6-13-2013

The Dynamic Multi-objective Multi-vehicle Covering Tour Problem

Joshua S. Ziegler

THE DYNAMIC MULTI-OBJECTIVE MULTI-VEHICLE COVERING TOUR PROBLEM

THESIS

Joshua S. Ziegler,

AFIT-ENG-13-J-09

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-13-J-09

THE DYNAMIC MULTI-OBJECTIVE MULTI-VEHICLE COVERING TOUR PROBLEM

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Science

Joshua S. Ziegler, B.S.

June 2013

AFIT-ENG-13-J-09

THE DYNAMIC MULTI-OBJECTIVE MULTI-VEHICLE COVERING TOUR PROBLEM

Joshua S. Ziegler, B.S.

Approved:

/signed/	21 May 2013
Gilbert L. Peterson, PhD (Chairman)	Date
/signed/	21 May 2013
Gary B. Lamont, PhD (Member)	Date
/signed/	21 May 2013
Maj Kennard R. Laviers, PhD (Member)	Date

AFIT-ENG-13-J-09

Abstract

This work introduces a new routing problem called the Dynamic Mulit-Objective Multi-vehicle Covering Tour Problem (DMOMCTP). The DMOMCTPs is a combinatorial optimization problem that represents the problem of routing multiple vehicles to survey an area in which unpredictable target nodes may appear during execution. The formulation includes multiple objectives that include minimizing the cost of the combined tour cost, minimizing the longest tour cost, minimizing the distance to nodes to be covered and maximizing the distance to hazardous nodes. This study adapts several existing algorithms to the problem with several operator and solution encoding variations. The efficacy of this set of solvers is measured against six problem instances created from existing Traveling Salesman Problem instances which represent several real countries. The results indicate that repair operators, variable length solution encodings and variable-length operators obtain a better approximation of the true Pareto front.

Table of Contents

Page

Page

List of Figures

List of Tables

List of Symbols

Symbol	Definition
c	covering distance
C^{capacity}	capacity for each vehicle, given as a scalar or vector when using multiple goods
d_i	demand for node $v_i \in V$; can be a scalar or vector when using multiple goods
D	vector of demands for all nodes $\forall v_i \in V \exists d_i \in D$
$e_{(i,j)}$	member of E - cost from node v_i to v_j
E	matrix of least cost paths from any node to any other node
$f_i(R)$	objective function i w.r.t. R
G	graph
k_i	vehicle i; $k_i \in K$
K	set of vehicles
L	maximum length of each vehicle route (assuming constant, identical speed; speed and cost condidered interchangable)
m_i	time slice i; $m_i \in M$
M	set of time slices
O	set of nodes that cannot be visited
P	subset of customers V that may demand goods later on (as in the Dynamic Travelling Salesman Problem)
r_i	route i; $r_i \in R$
R	set of all routes
T	set of nodes that must be visited
v_0	depot - starting and ending point of all vehicles (unless specified otherwise)
V^{all}	all nodes in G; superset of the more common $V^{\text{visitable}}$
$V^{\text{visitable}}$	all visitable nodes of G

Symbol	Definition
W	set of nodes that must be covered

List of Acronyms

Acronym	Definition
AbYSS	Archive-based Scatter Search
ACO	Ant Colony Optimization
ADP	Approximate Dynamic Programming
AI	Artificial Intelligence
AUV	Autonomous Underwater Vehicle
CLP	Clover Leaf Problem
CSP	Covering Salesman Problem
CTP	Covering Tour Problem
CVRP	Capacitated Vehicle Routing Problem
DAFP	Dial-A-Flight-Problem
DARP	Dial-A-Ride-Problem
DC	Divide and Conquer
DM	decision maker
DMOMCTP	Dynamic Mulit-Objective Multi-vehicle Covering Tour Problem
DOD	Degree of Dynamism
DREP	Distributed Resource Exploitation Problem
DTRP	Dynamic Traveling Repairperson Problem
DTSP	Dynamic Traveling Salesman Problem
EDOD	Effective Degree of Dynamism
EPCA	Exponential family Principal Components Analysis
FCFS	First Come First Serve
GIS	Geographic Information System
GRASP	Greedy Randomized Adaptive Search Procedures

Acronym	Definition
LW	Center-of-Gravity Longest Wait
mCTP	multi-vehicle Covering Tour Problem
MDP	Markov decision process
mDTRP	multi-vehicle Dynamic Traveling Repairperson Problem
MOEA	Multiobjective Evolutionary Algorithm
MOP	Multiobjective Problem
MOGA	Multi-Objective Genetic Algorithm
mSQM	multi-vehicle Stochastic Queue Mediam
mTSPTW	multi-Travelling Salesman Problem with Time Windows
NDP	Neuro-Dynamic Programming
NSGA	Nondominated Sorted Genetic Algorithm
NSGA-II	Nondominated Sorted Genetic Algorithm II
NN	Nearest Neighbor
NNRW	Nearest Neighbor with Random Weights
OP	Orienteering Problem
PACO	Parallel Ant Colony Optimization
PAES	Pareto Archived Evolution Strategy
PDP	Pickup and Delivery Problem
PDTRP	Partially Dynamic Travling Repairman Problem
POMDP	partially observable Markov decision process
PPP	Persistent Patrolling Problem
SPEA	Strength Pareto Evolutionary Algorithm
SPEA2	Strength Pareto Evolutionary Algorithm 2
RBX	Route Based Crossover
RH	Receding Horizon

Acronym	Definition
RP	Random Process
RS	Random Search
RV	Random Variable
RVRP	Robust Vehicle Routing Problem
SBRP	School Bus Routing Problem
SBX	Simulated Binary Crossover
TSP	Traveling Salesman Problem
UAV	Unmanned Aerial Vehicle
USBRP	Urban School Bus Routing Problem
VEGA	Vector Evaluated Genetic Algorithm
VRP	Vehicle Routing Problem
VRPTW	Vehicle routing problem with time windows

THE DYNAMIC MULTI-OBJECTIVE MULTI-VEHICLE COVERING TOUR PROBLEM

I. Introduction

Field intelligence quality and timeliness can be instrumental in war and peace. Recent advances in Unmanned Aerial Vehicles (UAVs) have provided an exciting new avenue for collecting that intelligence without risk to pilots. By allocating several UAVs to a given mission - cooperating as a swarm - mission time can be reduced. Planning such missions is a difficult problem however. The time required, unknown information, distance to surveillance targets, and distance to known hostile forces need to be considered. In order to study this problem, it must first be defined mathematically. This work uses the existing Covering Tour Problem (CTP) [65] as a starting point.

The CTP is a routing problem where a single vehicle must visit a subset of all the points provided while covering another subset [65]. In this context, the vehicle must visit some point A within a predefined distance of the node B to "cover" it. The objective is to minimize the cost of the route while abiding by these constraints. This basic formalizations can then be modified in order to study a number of real-world problems such as mobile medical unit routing [49, 78], school bus routing [124] and dial-a-ride routing [154].

This work takes the multi-vehicle Covering Tour Problem (mCTP) [73] variation of the CTP as a starting point because no existing variations of it - or any other routing problem - fit the native UAV problem described above. Namely, no existing problem exists that simultaneously provides dynamically revealed nodes, nodes to be avoided, covering constraints, multiple vehicles, and objectives to minimize the combined route cost, minimize the wall-clock time taken, maximized the combined distance to all nodes to avoid

and minimize the combined distance to all nodes to be covered. This work introduces such a problem formalization and studies a set of solution methods in an a posteriori fashion.

1.1 Contributions

This work introduces a novel CTP formalization - the Dynamic Mulit-Objective Multi-vehicle Covering Tour Problem (DMOMCTP) - which includes additional objectives and unknown nodes that appear during execution. The goal of the work is to find the best set of centralized solvers and operators from the tested set for the DMOMCTP. This allows future works to focus on promising algorithms while including the best from this work for comparison. This work also provides insight into operator design with respect to the DMOMCTP.

1.2 Background

Dozens of routing problem formalizations exist with varying constraints and objectives. While none of the individual components of the problem are unique, the combination presented here represents a new formalization. This work briefly reviews a number of related works on problems and their solution methods ranging from the Traveling Salesman Problem (TSP) [149] to the multi-vehicle Covering Tour Problem (mCTP).

1.3 Methodology

This work approaches the DMOMCTP from an a posteriori perspective, meaning that the result of any solver is a set of solutions. Furthermore, each solver is given three minutes to find the best solution set it can, as opposed to a set number of iterations. This models the time requirements of a real UAV system. Algorithms tested include Nearest Neighbor with Random Weightss (NNRWs), Archive-based Scatter Searchs (AbYSSs), Strength Pareto Evolutionary Algorithm 2 (SPEA2), Nondominated Sorted Genetic Algorithm II (NSGA-II) and Random Search (RS). Modified versions of these algorithm are also

included, which replace the operators and/or solution encoding of the base algorithm. Operators used include Simulated Binary Crossovers (SBXs), Route Based Crossovers (RBXs), Uniform addition, Uniform deletion, and monolithic repair. Solution encodings include fixed and variable length.

As is common to studies of this nature, the quality of these sets are assessed via several metrics [38]. This work uses hypervolume, generalized spread, and number of constraints violated. This complicates analysis because unlike single-objective studies, the best is subjective. The best ultimately relies on the preferences of the user. For example, a solver may have the best hypervolume values, but worst constraints violated values. Finally, although the multiobjective nature of the results and No Free Lunch Theorem limits the conclusions we can make from such experimentation, a rough ordering of the solvers tested is provided.

1.4 Limitations of Study

The native problem domain is inherently complex and difficult to solve which was the impetus for limiting the formalization's scope. Issues related to real-world problems such as navigation, communication, and physics are removed and left for later works. The resulting formalization is still an interesting and difficult problem that provides a stepping stone to more complete problems. Specifically, we have chosen to focus on central solvers which provide a solution baseline for future work.

Note that, this work only provides results for fifty independent runs for each algorithm over six problem instances - largely due to the real-world time required for computation. As in other Multiobjective Problems (MOPs) the nature of this problem and low number of data points, the resulting data is not normal. The problem instances themselves are taken from real-world Geographic Information System (GIS) data sets, but the nodes are assigned to the various sets using a Uniform Random Variable (RV) which is an assumption that may not hold in all problem instances.

1.5 Outline

Chapter 2 introduces a formal version of the native problem: the DMOMCTP. Chapter 3 reviews the routing problem literature from the computer science, operations research, and robotics communities. The DMOMCTP is shown to reduce to the TSP, making the DMOMCTP NP-complete. Chapter 4 defines the experiment, including the data and metrics chosen to measure algorithm efficacy with respect to the DMOMCTP. Chapter 5 provides and discusses the results.

1.6 Summary

Motivated by a complex, multiobjective, multivehicle routing problem with covering constraints this work introduces a new formalization - the DMOMCTP. Related works from routing problems, Artificial Intelligence (AI), and MOPs are discussed briefly. As a new problem formalization and given the No Free Lunch Theorem, these works provide limited value in solving this problem. Thus the purpose of this work is to study several centralized solvers against the DMOMCTP and provide a reduced set to recommend for future works and applications. Given the limited number of problem instances and independent runs, the work finds four promising algorithms including two versions of both NSGA-II and SPEA2. It then discusses the possible reasons for these results and future directions for research.

II. Problem Definition

This chapter introduces the unique problem domain - the Dynamic Mulit-Objective Multi-vehicle Covering Tour Problem (DMOMCTP) - to provide context before discussing related work in Chapter 3. While the DMOMCTP is related to other vehicle tour problems it differs in several details resulting in a fundamentally different problem.

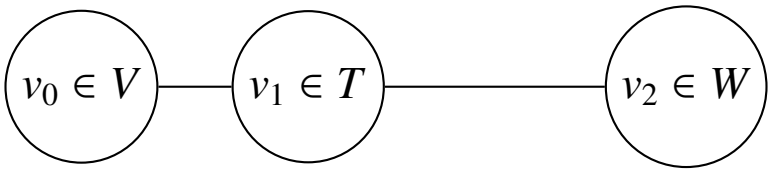


Figure 2.1: Third Objective Example.

Given an undirected graph $G = (V^{\text{all}}, E)$ where $\{v_0, v_i, \ldots, v_j\} \in V^{\text{all}}$ is the set of *all* nodes. E is a matrix representing the cost to travel between any two nodes such that $\exists e_{(i,j)} \in E \; \forall v_i, v_j \in V;\; e_{(i,j)} \in \mathbb{R} \geq 0;\; i \neq j$. The $e_{(i,j)}$ costs denote the cheapest path between any two nodes and not that the graph is fully-connected. This makes E a lossy representation of the true graph. The edges in the original undirected graph are symmetrical and satisfies the triangle inequality.

The nodes in V^{all} can belong to one or more of three sets: W, T and O which represent the nodes which must be covered (W), must be visited (T) and cannot be visited (O). The only restriction on set assignment is that nodes cannot belong to T and O simultaneously which would render the problem infeasible. Covering a node $v_i \in W$ requires at least one vehicle $k_n \in K$ to visit a node $v_j \in V;\; v_j \notin O$ within a preset distance c of the node to be covered $e_{(i,j)} \leq c$. This constraint can be represented formally as $\forall v_i \in W \exists v_j \in R \mid e_{i,j} \leq c$.

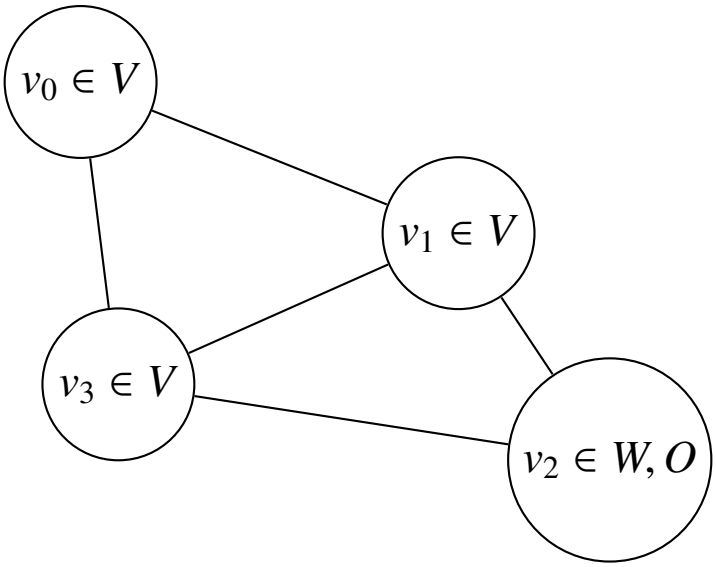


Figure 2.2: Covering Unvisitable Nodes.

The goal is to find a set of routes R (e.g. $r_0 = \{v_0, v_5, v_2, v_0\} \in R$) for K identical vehicles with infinite capacity, where $|R| = |K|$, to visit all nodes in T and cover all nodes in W without visiting any in O. All routes must begin and end at a preset node v_0 called the depot. Nodes can be visited any number of times by any number of vehicles.

Further complicating matters, the environment is dynamic in that previously unknown nodes are added to the graph as time progresses (i.e. environment is partially observable). These nodes are immediately assigned to their appropriate subsets (i.e. W, T, O). Nodes never disappear once known. The problem *can* be resolved after each after each state update, referred to as a time slice $m_i \in M$. Each time slice can be thought of as a static problem instance, however the vehicles K may start at any node $v_i \in V^{\text{all}}; v_i \notin O$ and may be between nodes.

The resulting set of routes R must meet the four objectives listed in Table 2.1. The first objective - $f_1(R)$ - is to minimize the combined cost of all routes (e.g. if cost is related to fuel consumption, this minimizes the amount of fuel used for the solution). The second

Table 2.1: DMOMCTP Objectives.

Shorthand	**Description**	**Mathematic formulation**
$f_1(R)$	minimize the sum of all routes' cost	$\min \sum_{r_n \in R} \sum_{i=0}^{i<\lvert r_n \rvert} e_{i,i+1}$
$f_2(R)$	minimize worst single route cost	$\min \max \sum_{r_n \in R} \sum_{i=0}^{i<\lvert r_n \rvert} e_{i,i+1}$
$f_3(R)$	minimize distance to all nodes covered	$\min \sum_{\forall v_i \in W} \min_{\forall v_j \in r_n \forall r_n \in R} e_{i,j}$
$f_4(R)$	maximize distance to all unvisitable nodes	$\max \sum_{\forall v_i \in O} \min_{\forall v_j \in r_n \forall r_n \in R} e_{i,j}$

($f_2(R)$) - minimize the worst route's cost - minimizes the cost of the entire solution when treating each vehicle as independent (e.g. if edge cost represents time, this minimizes the time taken to complete the problem). The third objective ($f_3(R)$) seeks to minimize the distance between each node to be covered and the closest visited node among all routes. This acts as an attractive force on the vehicle (e.g. if the nodes in W represent areas of interest to be investigated using on board sensors with a maximum range of c this increases the quantity of that sensor data).

For example, consider Figure 2.1. If v_0 is the depot, we could simply visit v_1 and meet the requirements (assuming the covering distance c is large enough to reach v_2). However $f_3(R)$ would equal the cost of $e_{(v_1,v_2)}$. If we instead visit both v_1 and v_2, $f_3(R) = 0$. This does increase the evaluation of the first two objective functions ($f_1(v_0, v_1, v_2, v_1, v_0) > f_1(v_0, v_1, v_0)$ and similarly for $f_2(R)$). Alternatively nodes may require a vehicle to cover them but cannot be visited (i.e. $v_i \in W$ and $v_i \in O$) as in Figure 2.2 with v_2 (in these cases, $f_3(R)$ cannot be minimized to 0).

The fourth objective ($f_4(R)$) is to maximize the distance between each node visited and each node in O. This acts as an repulsive force on the vehicles (e.g. if the nodes in O

represent hostile forces such as anti-aircraft installations, this reduces the risk our vehicles face).

III. Background

The Dynamic Mulit-Objective Multi-vehicle Covering Tour Problem (DMOMCTP) is composed of several components: dynamic nodes, multiple objectives, covering constraints and route planning. This chapter reviews previous work in these areas starting with the basic differences between single and Multiobjective Problems (MOPs). It briefly touches on basic solution methods and performance measures for each. It then proceeds to problem formulations, both single and multiobjective, which are related to the DMOMCTP. The final section of the chapter revisits solution methods, considering their pros, cons and past applications.

3.1 Problem Objectives

Problems can be split into two categories: single and multiobjective. Single objective problems determine the utility of their solutions with a single measure. For the Traveling Salesman Problem (TSP), the objective is to find the *shortest* route visiting each city exactly once and ending at the starting point. The utility of a TSP solution is its length. Single objectives problems can be defined mathematically as minimizing (or maximizing) the utility defined by some function $f(x)$ where x is the vector of decisions variables ($x = (x_1, \ldots, x_n)$) subject to any constraints. A constraint is any limitation on any decision variable(s) (i.e. the domain of the variable). For example, the TSP has the constraint that "each city may only be visited once."

3.1.1 Multiple Objectives.

Multiobjective problems differ in that their utility is defined as a vector. For example, if the TSP is modified to include multiple salesmen it may be desirable to minimize the variance among the assigned routes. Each solution's utility is then defined as a vector $f(x) = (k_1, k_2)$ which can be visualized in a two-dimensional objective space as

in Figure 3.1. Now consider two solutions: $s_1 = (10, 1)$ and $s_2 = (8, 2)$. Each solution is better than the other with respect to at least one objective meaning there is no single best [38]. The result from a MOP solver is a *set* of solutions.

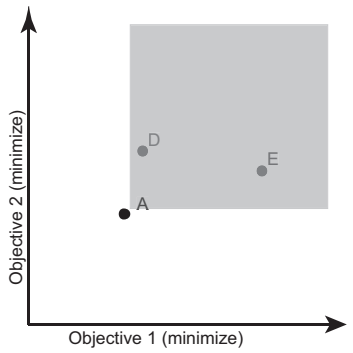


Figure 3.1: Pareto Dominance.

Formally, a solution s_i is said to *weakly* dominate another s_j if it has at least one objective $k \in K$ that is better than s_j's and is at least equal to s_j in all other objectives: $\forall k \in \{1, \ldots, k\}, f(s_i)_k \leq f(s_j)_k \wedge \exists l \in \{1, \ldots, k\} \mid f(s_i)_l < f(s_j)_l$ [38]. A solution s_i is said to *strongly* dominate another s_j if it improves upon all objectives: $\forall k \in \{1, \ldots, k\}, f(s_i)_k < f(s_j)_k$. For example, solution A strongly dominates both D and E in Figure 3.1 because all of its objectives are better than theirs. More generally, A would strongly dominate any solutions in the light gray box. Solution A is said to be non-dominated.

The set of solutions which are not strongly dominated (but may be weakly-dominated) is known as the Pareto front, which is a subset of the entire solution set. For example, solutions A, B, and C in Figure 3.2 are non-dominated and form the Pareto front out of the solution set $S = \{A, B, C, D, E\}$. For every MOP, there exists a Pareto front known as P_{true} which is the best possible set of non-dominated solutions. Typically this set is infeasible to compute and can only be approximated [38]. This approximated Pareto front is the best *known* Pareto front and is denoted P_{known}.

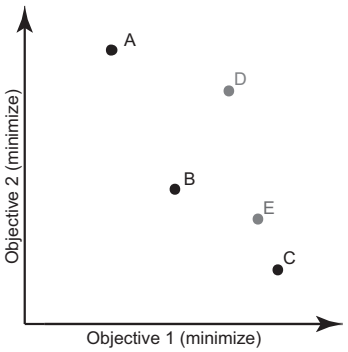


Figure 3.2: Pareto Front.

3.1.2 Solving Multiobjective Problems.

Algorithms for solving multiobjective problems are categorized here into two groups: deterministic and stochastic solvers. Enumerative methods are a special type of deterministic method. Enumerative methods are those which evaluate all possible solutions given the problem constraints [38]. These methods are simple to implement and understand but become impractical when the search space becomes large [38].

Other deterministic methods exist which exploit domain knowledge in the form of heuristics to circumvent the search space explosion [14]. Examples include greedy search, hill-climbing and branch and bound. However, real-world MOPs are "high-dimensional, discontinuous, multimodal, and/or NP-Complete ... problems exhibiting one or more of these above characteristics are termed irregular. Because many real-world scientific and engineering MOPs are irregular, enumerative and deterministic search techniques are then unsuitable" [38].

Stochastic algorithms form the second type of algorithms. Examples include simulated annealing, Monte Carlo simulation and evolutionary computation. This class of algorithms use random decisions in an attempt to approximate the best solution(s). Perhaps the most basic stochastic algorithm is a random walk. In a TSP instance, a Random Variable (RV) would iteratively choose the next node to visit and remove it from the list

of candidates. A Random Variable (RV) is a value is decided by chance in a mathematical sense. That is, it does not have a set value. When the list of candidates is empty the solution is evaluated and the process restarts. Many stochastic algorithms use problem domain-specific heuristics or operators to guide their search to strategically search the space.

3.1.3 Decision Maker.

Since multiobjective problem solvers result in a set of solutions, there must be a human decision maker (DM) to choose "the best" [38]. This can be done one of two ways: a priori or a posteriori.

A priori methods require the human DM to choose a method of combining the objectives into one [38]. There are several ways to do this, perhaps the simplest of which is to create a weighted combination function (e.g. $f(s) = a \cdot f_1(s)^b + \cdots + c \cdot f_k(s)^d$) [38]. The multiobjective problem can then be treated as single objective.

A posteriori methods require a human DM to choose a solution from the set. The objectives are not altered or combined leaving the DM to determine which trade offs are best [38].

A priori methods are desirable for their relative simplicity. However, human DM preferences are difficult to define mathematically, or are problem dependent in practice [38]. A posteriori methods are simple in theory, but presenting the solutions to the human DM can be a difficult data visualization problem in itself [26].

3.1.4 A posteriori Performance Measures.

Since a posteriori solvers produce a set of solutions, the quality of that set must be measured. Desirable qualities of a solution set include the degree to which it approximates P_{true} and the spread or variety of the solutions along the Pareto front [38]. As discussed above, P_{true} can be infeasible to compute, so many a posteriori multiobjective works instead use a computed set called P_{known} for comparison [38]. This set can be created by

aggregating all non-dominated solutions from all algorithms and represents the best known Pareto front [38]. The following sections briefly cover several metrics.

3.1.4.1 Hypervolume.

$$\text{HV} = \{\bigcup_{i} \text{area}_i \mid \text{vec}_i \in P_{\text{known}}\} \tag{3.1}$$

Hypervolume is a metric that measures the volume represented by all of a set's members in n-dimensional objective space (i.e. hyperspace) [38]. This can be provided as a ratio of P_{known}'s hypervolume. A solution set's hypervolume is the summation of all of the hypercubes' volume. These shapes are bounded by each solution's fitness and an arbitrary boundary point (e.g. the worst solution) in objective space [38]. The formula for hypervolume is shown in Equation (3.1) where "vec_i is a non-dominated vector in P_{known} and area_i is the area between the origin and vector vec_i [38]. If the front of non-dominated solutions is not convex, the resulting hypervolume may be misleading [38]. This metric is to be maximized and provides a sense of the sets goodness, but not its diversity [38]. A hypervolume ratio of 1 would mean that the set fills up a space as great as the known Pareto front.

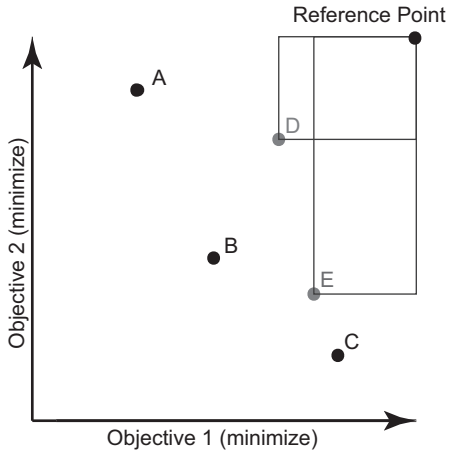


Figure 3.3: Hypervolume Example for Two Objectives.

Figure 3.3 presents a two dimensional example. The points A through E represent solutions A through E in objective space. If Algorithm A produces the non-dominated set (D, E) its hypervolume would be calculated by first choosing a reference point (see top right) and then calculating the area covered by the bounding boxes from each point to the reference point. This combined area is the hypervolume of algorithm A's non-dominated solution set. Repeating this process for the known Pareto front, we then come up with the ratio of algorithm A's hypervolume to the known Pareto front ($0 \leq HV \leq 1$).

3.1.4.2 Generational Distance.

$$GD = \frac{(\sum_{i=1}^{n} d_i^p)^{\frac{1}{p}}}{|PF_{\text{known}}|} \tag{3.2}$$

Generational distance - see Equation (3.2) - is a measure of the average distance between a non-dominated set's members and those of the known Pareto front in objective space (n-dimensional) [38]. PF_{known} is "the number of vectors in PF_{known}, $p = 2$, and d_i is the Euclidean phenotypic distance between each member, i, of PF_{known} and the closest member in PF_{true} to that member." In our experiment, PF_{known} is the non-dominated set from an algorithm, and PF_{true} is the non-dominated set of solutions created by aggregating all solutions from all runs and algorithms which we refer to as P_{known}. This metric is to be minimized.

3.1.4.3 Generalized Spread.

Generalized spread measures the distance from a given point to its nearest neighbor. This measure was introduced by Durillo and Nebro [52] as a replacement for the spread metric which only works in 2-dimensional space. Mathematically:

$$\text{Spread} = \frac{\sum_{i=1}^{m} d(e_i, S) + \sum_{X \in S} |d(X, S) - \bar{d}|}{\sum_{i=1}^{m} d(e_i, S) + |S| * \bar{d}} \tag{3.3}$$

where "S is a set of solutions, S^* is the set of Pareto optimal solutions $(e_1, \ldots, e_m)$ are m extreme solutions in S^*, m is the number of objectives and"

$$d(X, S) = \min_{Y \in S, Y \neq X} ||F(X) - F(Y)||^2 \tag{3.4}$$

$$\bar{d} = \frac{1}{|S^*| \sum_{X \in S^*} d(X, S)} \tag{3.5}$$

Generalized spread measures the diversity of the solutions set and is to be minimized.

3.1.4.4 Constraints Violated.

Number of constraints violated is not a MOP specific metric, but provides an important measure of solution feasibility for constrained problems like the DMOMCTP. This is useful because many algorithms allow for infeasible solutions in an attempt to circumvent local optima to find global optima. The calculation of violated constraints is problem domain specific. This metric is to be minimized.

3.2 Related Problems

The DMOMCTP is, at its core, a routing problem. Given the popularity and numerous variations of such problems, the body of previous work is expansive (especially in the operations research community) [42]. Since the DMOMCTP is a new problem formulation we briefly review a broad range of related works. Particular focus is placed on the multi-vehicle Covering Tour Problem (mCTP), Covering Tour Problem (CTP) and TSP, which are the most closely related problem formulations.

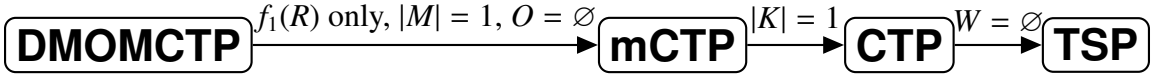


Figure 3.4: DMOMCTP relations to the m-CTP, CTP and TSP.

To understand why the focus is placed on these three problems, this section briefly outlines their relation to the DMOMCTP informally (shown in Figure 3.4). Ignoring the dynamic and multi-objective elements of the DMOMCTP (i.e. using $f_1(R)$ from Table 2.1 for our singular objective) we are left with the mCTP. Utilizing only one vehicle in that problem reduces it to the CTP. Removing the covering constraint from the CTP reduces the problem to the TSP. Thus the DMOMCTP is related to these three problems which explains our detailed coverage. Works that differ in more significant ways (e.g. such as changing the objective function) are covered in less detail, relying instead on applicable reviews when possible.

The most basic formulations of these problems - as indicated above - have been static, deterministic, single-objective versions of the TSP, CTP and Vehicle Routing Problem (VRP). Reviews of work in this area include [16, 39, 89, 90]. Recent developments enabling real-time communication, global positioning and re-routing has resulted in research in dynamic, partially-observable and stochastic variations of these problems [42, 66, 126]. Given the rise in unique problem formulations and complex relations between them, we have broken out many of the constraints and characteristics associated with each variation in Table 3.1. The sections that follow review these elements and the problems that include them.

3.2.1 Base Routing Problem (the Traveling Salesman Problem).

Possibly the most well-known of routing problems is the TSP [55, 149]. Given a graph $G = (V, E)$ where V are the nodes and E the matrix of edge costs, the problem is to find the shortest path originating at the depot, visiting all cities (i.e. $\forall v_i \in V$) and then returning to the depot. Lawler, et al. [94] provides an extensive overview of the TSP. The problem was shown to be NP-hard by Karp [87]. This is important because it forms a special case of the DMOMCTP where there is only one vehicle ($|K| = 1$), no nodes to cover ($W = \varnothing$), no nodes that cannot be visited ($O = \varnothing$), only one time slice $|M| = 1$, all nodes belong to

Table 3.1: Common routing problem elements.

Element	Description	Papers
Backhauls	customers split into two sets: deliveries and pickups	[41, 112, 133]
Capacitated	each customer has a scalar demand for a given good and the vehicles have finite capacity	[45]
Covering	another set of nodes are required to be covered, which requires at least one vehicle to visit a customer within a specified distance of that customer	[44, 146, 159]
Dynamic	the environment is partially-observable and may change over time (e.g. revealing new customers, new demand, etc.)	[23–25, 30, 51, 53, 54, 57, 59, 83, 125, 126, 140, 141, 143, 152]
Heterogeneous fleet	vehicles differ in capacity, speed or some other characteristic	[36]
Max/Min tour cost	added constraint that the total tour cost must adhere to the minimum or maximum; often coupled with the profits variant	[12, 15]
Multiple depot	instead of a single depot vehicles may begin and end at any of the depots	[47]
Multiple objectives	the problem requires a priora or a posteriori optimization for several objective functions instead of just one	[27, 49, 143, 147]
Multiple products	customer demand is expressed as a vector rather than a scalar representing demand for different products; vehicles have a corresponding vector of capacity and current load	[45]
Multiple vehicle	K vehicles are available; note $\|K\| = 1$ is the TSP	[34, 45, 59]
Profits	each customer is associated with a profit which must be maximized and not all customers need to be visited	[31, 55]
Time windows	each customer must be visited during a specific time frame	[40, 47]
Underdelivery	a vehicle can supply a customer with its remaining goods even if it does not meet their full demand	[45]

the must visit set ($T \subseteq V$) and only $f_1(R)$ is used (single objective). Thus, the DMOMCTP is also at least NP-hard and solution methods for the TSP may show promise when adapted to the DMOMCTP. Especially when considering works that capture more elements of the DMOMCTP (e.g. dynamic nodes, multiple-objectives, covering constraints, etc.).

3.2.2 Covering Constraints.

Covering constraints are used in the CTP [65] where nodes in the set W must be within some radius c of a visited node. The problem is otherwise the same as the TSP where its objective is to minimize the total route cost ($f_1(R)$) with a single vehicle ($K = 1$) while

visiting all of the nodes in T (a subset of V in the CTP). This is useful for modeling things such as populations which must be served, as in the School Bus Routing Problem (SBRP) for which [124] provides a review of the various formulations of the native problem (e.g. some consider bus stops as being defined a priori, others create them as a part of the problem, etc.). The SBRP includes multiple vehicles ($K \geq 2$) and typically includes constraints on the vehicle's capacity (i.e. number of seats) and route length.

Doerner, et al. [49] extends the concept of covering to a multi-objective version of the VRP. Jozefowiez, et al. [85] study a bi-objective CTP which minimizes the total route cost ($f_1(R)$) and the distance from the nodes to be covered to those visited ($f_3(R)$). This bi-objective version differs from the DMOMCTP in that it does not include two objectives (i.e. $f_2(R)$ and $f_4(R)$), is static and only uses one vehicle ($K = 1$).

Current and Schilling [44] present the Covering Salesman Problem (CSP) which incorporates covering constraints in a different fashion. The objective of the CSP is to craft a "minimum cost tour of a subset of n given cities such that every city not on the tour is within some predetermined covering distance."

3.2.2.1 Multiple Vehicle Covering Tour Problem.

Closer to our problem is the mCTP as defined in [73] which is based on the original CTP formulation by [65]. This problem is a generalization of the static CTP which includes additional vehicles ($K \geq 2$). Additional constraints may also be included as in the formulation by Hachicha, et al. [73] which constrains each customer in V to be on at most one route and those in T must be on exactly one route while limiting the number of customers on each tour (constant p) and limiting the maximum individual route cost (constant q). Note Hachicha, et al. defined the graph as $G = (V \cup W, E)$ where V is the set of customers that may be visited, T is the subset of V that must be visited and W the set that must be covered. The formulation by Hodgson, et al. [78] has $V \subseteq W$ and does not

include a subset that must be visited ($T = \varnothing$) thus reducing the graph to $G = (W, E)$ (i.e. all nodes must be covered and all may be visited).

Work by Hachicha, et al. [73] describe three heuristics for the mCTP: one based on the savings heuristic, another based on the sweep heuristic and a third being a route-first, cluster-second method. They found the modified savings heuristic to be the fastest but the modified sweep and route-first, cluster seconds methods to be better with respect to solution quality.

Wolfler and Cordone [156] introduce a problem bearing some similarity to the mCTP which they term the overnight security service problem. The problem is to decompose the geographical space into m clusters to be serviced by m guards. Each individual guard must then visit the customers within their cluster. There are also multiple demand types. The covering constraint in the problem models the guard's range in which they can swiftly address a burglar alarm. The guards must remain within that distance to all of their assigned nodes (alarmed buildings within the cluster) while carrying out their tour (servicing the customers within their cluster). It can be seen as a combination of a clustering problem and multi-Travelling Salesman Problem with Time Windows (mTSPTW). The problem's native objectives include minimizing the costs, maximize service, minimize response time to alarms, minimize variance in task distribution among guards, and to allow for variety of routes such that burglars cannot predict the guard's schedule. The formalization presented only uses a single objective which utilizes a weighted combination function to encode all of the native objectives.

Bowerman, et al. [27] provide one of the only attempts to study the closely related multi-objective Urban School Bus Routing Problem (USBRP). That formulation of the USBRP includes objective functions to minimize the number of routes, minimize the route lengths ($f_1(R)$), minimize variation in the number of students in each route, minimize the variation in the route lengths, and minimize the student's walking distance. They also

consider several side-constraints such as an upper bound on the number of students on each route (i.e. capacitated vehicles) and an upper bound on the travel time of each route (i.e. time windows). This problem differs from the DMOMCTP in that it is static, uses different objectives (the only common objective between them is $f_1(R)$) and has capacitated vehicles.

Enright, et al. [60] consider a problem called the Persistent Patrolling Problem (PPP) which - in their case - models multiple Unmanned Aerial Vehicle (UAV)s with limited sensor range (the covering distance) and nodes that are dynamically added in response to vehicle movement. The problem is otherwise modeled as the Dynamic Traveling Repairperson Problem (DTRP). The DTRP can be thought of as a dynamic problem defined over a bounded Euclidean plane where nodes appear as time goes on. The objective is to minimize the time between when they appear and when they are serviced with multiple omnidirectional vehicles, each traveling at a constant velocity. As common with the DTRP Enright, et al. [60] investigate policies for solving the PPP which in the case of the DTRP, policies define the next action deterministically with respect to the current state.

3.2.3 Dynamic Problem Variants.

Wilson and Colvin's work on the Dial-A-Ride-Problem (DARP) [154] introduced the notion of the dynamic vehicle routing problem [126]. The DARP requires a set of vehicles to pickup customers and drop them off at arbitrary locations, both of which are revealed over time (rather than a priori). Later, Psaraftis [130] introduced the concept of an immediate request: those which required immediate replanning such that it is serviced as soon as possible.

Common dynamic elements in vehicle routing include known customers changing their demands for goods and services [11, 18, 20, 23, 63, 68, 70, 75, 79–81, 92, 110, 113, 151], adding previously unknown customers [120, 121, 137, 138], changing edge costs (e.g. due to congested traffic) [17, 32, 57, 71, 74, 101, 129, 145, 148] and vehicle availability (e.g. due to vehicle breakdown) [96, 97, 115, 116]. Some works provide the underlying

RV for these dynamic elements or allow sampling of that RV [126]. Reviews of dynamic routing problems and solution methods include [22, 64, 66, 67, 82, 126].

Dynamism complicates these routing problems in relation to their static version by increasing the degrees of freedom [126]. The future affects solution quality but cannot be observed a priori. Lund, et al. [102] denotes this added complexity as Degree of Dynamism (DOD). A problem's DOD is the ratio of the number of immediate requests in relation to the total number of requests. Because the timing of the reveals is also important, Larsen [91] introduced Effective Degree of Dynamism (EDOD) which is presented in Equation (3.7). Let M be the full time period requiring planning, V be the set of requests and m_i be the time request $i \in V$ is revealed. EDOD "represents the average of how late the immediate requests are received compared to the latest possible time these requests could be received" and thus provides a better measure of dynamism than DOD [91]. EDOD is defined over a fixed interval: $0 \leq d_2 \leq 1$. Fully static problems would have a $d_2 = 0$ (i.e. all requests are known a priori) and fully dynamic would have an $d_2 = 1$.

$$DOD = d_1 = \frac{n_{\text{immediate requests}}}{n_{\text{total requests}}} \tag{3.6}$$

$$EDOD = d_2 = \frac{\sum_{i=1}^{n_{\text{immediate}}} (\frac{m_i}{M})}{n_{\text{total}}} \tag{3.7}$$

As noted by [93], re-optimizing after every update may be computationally impractical except for problems with a low DOD. More generally, solution method performance may differ with respect to DOD.

3.2.3.1 Dynamic Demand for Known Customers.

A number of works address the problem of known customers which have unknown, or partially known demand. These works have developed a number of methods for adapting

and/or anticipating future demands. Ghiani, et al. [68] use anticipatory insertion, and sample-scenario planning.

In anticipatory insertion [68], a tour is constructed by inserting as many nodes requiring service as possible. Then means that it iterates over the set of customers choosing the largest serviceable set possible given the time limit. If a tie occurs, the one with the least costly path is selected. The initial tour is constructed using the method suggested in [106] which is to include all possible customers in T. The first node in T is used as the start, with the last being the ending node and the remaining nodes in T being added via Greedy Randomized Adaptive Search Procedures (GRASP) [56] with a savings criteria and post-processing variable neighborhood search algorithm using 1-shift and 2-opt as neighborhoods. Both methods also use the Center-of-Gravity Longest Wait (LW) heuristic from [151] which chooses whether to have the vehicle wait at its current location or to move on. This is important as it may be able to sit near a cluster of nodes and quickly service them should they require it.

The sample-scenario planning method [68] implements a modified version from [21] (which applied it to the Vehicle routing problem with time windows (VRPTW)). This approach maintains a set of routes with the set modeling the possible scenarios in which the nodes not currently requesting service might do so later. To do this, they use greedy insertion of new requests and model the problem as an Orienteering Problem (OP). The OP uses multiple vehicles K, a time limit L, and each node has an associated point value that it is worth. The problem is a routing problem with the time windows and profits element where the objective is changed to maximize those profits. They then use a consensus function to decide the most likely scenario and apply it using the least commitment strategy.

3.2.3.2 Adding Previously Unknown Customers.

Problems - such as the DMOMCTP - that add previously unknown customers during execution (i.e. partially observable environment) fall into two groups: those that reveal

them based on vehicle movement/sensors (e.g. path-finding in unknown terrain) and those that reveal them irrespective of the vehicle movement (e.g. DTRP). The DMOMCTP belongs to the latter group. O'Rourke [122] includes adding previously unknown customers for the VRP and Dynamic Traveling Salesman Problem (DTSP). Doumit and Minai [51] approach a variation of the VRP and TSP called the Distributed Resource Exploitation Problem (DREP) which also features previously unknown customers.

Methods vary for handling this type of state update. O'Rouke, et al. [122] treat such nodes as high priority targets and direct the singular UAV to them. The solver would then resolve the problem with that priority node as the starting point for the route.

To our knowledge adding previously unknown customers has never been included in work on the CTP or mCTP. Routing problems that do include this element include the DTRP, DARP, Dial-A-Flight-Problem (DAFP), and path finding in unknown terrain. However the DTRP, DARP and DAFP all differ from the DMOMCTP in their objective function(s).

3.2.3.3 Dial a Ride/Flight Problem.

The DARP features dynamic requests (from people) which must be picked up and dropped off; with the requests and their locations being revealed over time (the DAFP is the air transport version) [150]. The problem can be thought of as a more general version of the Pickup and Delivery Problem (PDP). The first works approaching the problem by Wilson, et al. [150, 154, 155] utilized insertion heuristics to handle the revealed customers. Madsen, et al. [104] used a similar method for a multi-objective version that included time windows and multiple capacities. Work by Gendreau, et al. [63] approaches a similar problem with soft time windows and multiple objectives (reduced to a single objective via an a priori weighting function), that adapts Tabu search with adaptive memory to the problem. Herbawi and Weber [76] compared ant colony optimization to a modified version of NSGA-II to solve a multi-objective variant which modeled multi-hop ridesharing. The

comparison to NSGA-II was based in part on their work in comparing genetic algorithms to the same problem in [77]. The objectives in both works were to minimize cost, time and the number of drivers required. Waisanen, et al. [153] study several policies under limited communication constraints. Chevrier, et al. [33] present a hybrid evolutionary method to solve a multiobjective version of the DARP.

3.2.3.4 Dynamic Traveling Salesman Problem.

Given the degree to which the TSP differs from the DMOMCTP, we cover works that focus on the DTSP briefly. Ghiani, et al. [69] present policies (deterministic functions with respect to the current state) for addressing the problem. The policies Ghiani, et al. present include two which segment the space and then use tours within each segment, and one that uses a single tour and unsegmented space. The two that use segmented space differ in whether they service newly revealed customers in the current tour (if they appear in the segment currently being serviced) or if they service them in a later tour. The third uses an insertion function that chooses the new customer's position in the tour by minimizing the insertion cost.

Li, et al. [95] present an Inver-Over algorithm [149] that utilizes a gene pool for the problem (holding the most promising edges) which can been seen as a heuristic based on genetic algorithm principles. Eyckelhof, et al. [54] presented an Ant Colony Optimization (ACO) solver for handling dynamic edge costs. Mavrovouniotis and Yange [107] presented a memetic ACO algorithm for a variation of the TSP in which revealed new cities could replace old ones (and the edges connecting them). Li and Feng [100] and Yang, et al. [157] present parallel algorithms for a dynamic, multiobjective version of the TSP. Ravassi [131] studies several policies (as defined here as deterministic functions with respect to the current state) adapted from Nearest Neighbor (NN) policy common in DTRP research. These policies are applied to a version of the TSP where the revealed locations and times of

customers are according to a Poisson process and are independent and uniformly distributed in time within a bounded Euclidean plane (similar to the DTRP).

3.2.3.5 Dynamic Traveling Repairperson Problem.

The DTRP was defined by Bersimas and Ryzin [23] and later expanded to include multiple vehicles by the same authors [24]. The DTRP is defined over a bounded Euclidean plane with a single omnidirectional vehicle that travels at a constant velocity. Demands are inserted dynamically. These service locations arrive in time according to a Poisson process with rate λ, and their locations are independent and uniformly distributed in the plane. The service location's demand (in time) are independent and identically distributed with $\mu = \bar{s}$ and $\mu_2 = \bar{s^2}$ (the mean and variance, respectively) with $\bar{s} > 0$. Locations cannot be serviced preemptively (before they are known). The objective is to find a *policy* that minimizes the waiting time nodes experience with respect to their arrival time. This deviates from the typical objective of routing problems to minimize the travel cost and has lead to the problem being viewed from a queuing theory perspective.

Table 3.2: DTRP Policies Proposed by Bertsimas and Van Ryzin.

Name	**Description**
First Come First Serve (FCFS)	vehicles serve each customer in the order it appeared; when no customers require service the vehicles remain where they are
Stochastic Queue Median (SQM)	specilization of FCFS where vehicles wait in the median of the Euclidean plan until a customer requires service
Nearest Neighbor (NN)	after servicing a customer the vehicle services the nearest customer to its current location
Partitioning policy (PART)	Euclidean plan is partitioned into smaller regions with vehicles assigned to each; each vehicle uses FCFS for servicing the nodes in its subregion

Several policies have been suggested and tried - via simulation or proof - though these policies usually vary in performance with respect to the so-called load type. Bertsimas and Van Ryzin proposed several of the most studied policies which are summarized in Table 3.2 [23]. For example, [24] found the multi-vehicle Stochastic Queue Mediam (mSQM)

policy to be asymptotically optimal for light load instances for the multi-vehicle Dynamic Traveling Repairperson Problem (mDTRP). Heavy load instances are more difficult and the lower performance bound is not known to be tight [140]. Results from [23, 25] have shown the NN policy to perform as well as the best known policy over the full range of instance types (i.e. light, medium and heavy) for the single vehicle case. However Shin and Lall [140] have improved on this work by introducing an Approximate Dynamic Programming (ADP) based policy which was shown to be better than the NN policy in the light load case for the mDTRP by proof. In the medium and heavy load cases it was shown to perform as well as or better than the NN by simulation (for one and three vehicle instances). The ADP policy is notable because it performs well over all load types, is decentralized, requires less computation than other policies and adapts to environmental changes [140].

Other works on DTRP variations include [59] which presented a decentralized policy which was shown via proof to be optimal with respect to the mDTRP light load case. The policy also performed well in the heavy load case, but results were limited to a single vehicle and the policy requires vehicles to know the location of "all other vehicles with contiguous Voronoi cells" in real time. Arsie, et al. [10] presented work on a similar problem which requires no communication among the agents. Smith, et al. [141] studied the single-vehicle variant with two types of priority demands (high and low). The objective of that formulation is to minimize a weighted function of the two queues average waiting time. Pavone presented two policies, Divide and Conquer (DC) and Receding Horizon (RH) for the DTRP. They were then extended to the mDTRP via distributed partitioning algorithms [125]. Irani, et al. [83] present two algorithms for the DTRP with deadlines.

Other variations of the DTRP exist, include the Partially Dynamic Travling Repairman Problem (PDTRP) as defined by [91] which includes a subset of customers which are not dynamic and changes the objective to minimize the total path cost as in the TSP.

3.2.4 Multiple Objective Variations.

Multiple objective variations of routing problems have become increasingly prevalent in the literature for which Jozefowiez, et al. [86] provide an excellent review of the VRP. The motivation is to include other realistic objectives in addition to minimizing combined path cost (i.e. $f_1(R)$ from Table 2.1). Objectives studied include driver workload balancing [27, 98], customer satisfaction [139], number of vehicles used [118, 139], robustness [109] and covering [49].

3.2.5 Multiple Vehicles.

The VRP, a generalization of the TSP, was defined by Dantzig and Ramser [45]. The VRP is defined on a graph $G = (V, E, C)$, where $V = \{v_0, \ldots, v_n\}$ is the set of customers, $E = \{(v_i, v_j) \in V, i \neq j\}$ is the set of edges connecting those customers, and $C = (c_{(i,j)})_{(v_i,v_j)\in E}$ is a cost matrix defined for all $e_{(i,j)} \in E$. Each customer pair is defined in the cost matrix C even if no direct edge $e_{(i,j)} \in E$ exists between them. Instead the matrix values represent the cheapest route between any two customers. The matrix E is symmetric and satisfies the triangle inequality. Typically v_0 is defined as the sole depot: the origin and final destination for all vehicles. The objective is to then find a set of routes for K identical vehicles such that each customer is visited exactly once while minimizing the sum of the routes cost.

Unlike the DMOMCTP the VRP has only one time slice (i.e. $|M| = 1$), no nodes to cover (i.e. $W = \varnothing$), no nodes which cannot be visited (i.e. $O = \varnothing$) and only one objective: $f_1(R)$ (to minimized the combined cost of all routes). It also adds constraints such that each customer may only be visited once over all routes, and vehicles have finite vehicle capacity C^{capacity} (with corresponding customer demand $d_i \in D$) whereas the DMOMCTP does not included capacitated vehicles.

The original [45] paper also mentions several variations including route cardinality constraints, multiple products, and partial fulfillment of orders (i.e. underdelivery). The cardinality constraints were given as a limit of m nodes, m being a divisor of the $n - 1$

remaining customers (the Clover Leaf Problem (CLP)). Multiple products allow for customer demand to be expressed as a vector instead of a scalar with corresponding vectors specifying vehicle capacity and current load. Partial fulfillment addresses the case where a vehicle has capacity left before returning to the depot, but no customer which it would fully sate. When the VRP only includes capacitated vehicles and not route length constraints (or any other wrinkles) it is typically denoted as Capacitated VRP [39].

Work on the static, deterministic, single-objective VRP has been extensive. While exact solution methods have been studied, they remain ineffective on instances above 100 customers resulting in heuristics and metaheuristics being used most often in practice [90]. The type of metaheuristics have run the gamut, from tabu search, simulated annealing and genetic algorithms to ant colony optimization. Solid reviews include [16, 39, 89, 90]. Other works include Sorensen and Sevaux [143] who applied a Monte Carlo-based metaheuristic to several dynamic variants of the Capacitated Vehicle Routing Problem (CVRP).

Related work exists in the field of robotics with an emphasis on solutions that can be decentralized and require little to no communication or central solver [30, 48, 88, 114].

The mCTP - while already have been covered in Section 3.2.2.1 - is yet another problem featuring multiple vehicles.

3.3 Solution Methods

As a new problem formulation, there are no previous works addressing solution methods for the DMOMCTP. However this section briefly review methods used in many of the routing problems described above. Table 3.3 presents a full listing. Some methods are difficult to categorize due to their hybrid nature and the subjective nature of such categories.

3.3.1 Exact Methods.

Early work in problems such as the TSP relied on exact methods: those which logically consider every possibility while potentially ruling others out [37, 45, 105]. Given that the

Table 3.3: Common solution methods for routing problems.

Method Type	Sub-Type	Works
Exact methods	Misc.	[13, 37, 55, 65, 72]
Heuristics	Misc.	[22, 27, 28, 31, 32, 41, 44, 47, 55, 65, 73, 79, 156]
	Column Generation (CG) algorithms	[36, 40, 47]
	Market-based methods	[48]
	Cooperative Receding Horizon (CRH) algorithms	[34]
	Neighborhood search	[70]
	Policies	[10, 23–25, 30, 53, 60, 69]
Metaheuristics	Ant Colony Optimization (ACO)	[49, 50, 54, 76]
	Genetic algorithms	[8, 20, 33, 35, 49, 61, 62, 74, 76, 77]
	Tabu search	[9, 11, 11, 18, 29, 43, 50, 63]
	Multiple Scenario Approach (MSA)	[21, 68]
	Regret algorithms	[19]

DMOMCTP is at least NP-hard (reduces to the TSP) and because this work approaches the problem from an a posteriori multiobjective perspective exact methods are assumed to be ineffective in realistic problem sizes and focus is instead placed on approximation methods.

3.3.2 Heuristics.

First, note "that no agreed definitions of heuristics and metaheuristics exist in literature, some use heuristics and metaheuristics interchangeably" [158]. For this work a slightly different but useful definition of both heuristic and metaheuristic are used which define them based on the type of information they rely upon. Specifically this work defines heuristics as problem domain specific approximation methods that require more than a black-box model to operate. This is in contrast with metaheuristics which only require a black box model to operate (but many real applications use problem-specific heuristics when creating tailored version for use). Put another way, heuristics attempt to exploit features of the problem to find good, but not optimal solutions.

Heuristics work well for single objective problems due to the relative ease in which an algorithm designer can craft such heuristics. For multiobjective problems, such heuristics may function as intended for a single objective but can have unintended side-effects

with respect to the other objectives. For example, we may use a TSP heuristic for the DMOMCTP to decrease $f_1(R)$ but this likely worsens $f_2(R)$. Similarly, we may use potential field control methods to guide the vehicles among the nodes to avoid (O) and those to cover (W) but the algorithm may get stuck in a local optima.

Works of note utilizing heuristics include Bowerman, et al. [27] who used problem decomposition and heuristics to solve a multiobjective version of the USBRP.

3.3.2.1 Extracting Most Likely State.

One candidate solution method involves extracting the most likely state from the agent's state model (e.g.probability distribution, etc.). As Russell and Norvig state, this works well "when uncertainty is small" [135]. In problems where the state transitions are uncertain but its state is fully observable the problem can be modeled as a Markov decision process (MDP). MDPs are then solved resulting in a policy, providing the optimal action given any state. When the problem is partially observable - as in the DMOMCTP - the problem can be modeled as a partially observable Markov decision process (POMDP). The solution to which is also a policy, however it is not defined with respect to the state, but the agent's belief state. In practice POMDP methods are computationally intractable [103, 123]. Approximation methods have been presented to address this issue with promising results such as point-based value iteration [127]. Other methods include using Exponential family Principal Components Analysis (EPCA) to reduce the high-dimensional belief spaces [134].

This method may not be particularly useful for the DMOMCTP because the unknown information (what nodes will exist in the future and when they will be revealed) is not correlated to the agent's actions, only its current time.

3.3.2.2 Robust Control.

An alternative is robust control methods which assumes a bounded amount of uncertainty and do not assign probabilities to values within that interval. Robust solutions

work regardless of the realized value as long as it exists within the assumed bounds [135]. Such methods accept uncertainty and instead formulate solutions that can still succeed given the parameters.

Works that use this method for vehicle routing include [144] which looks at the CVRP with dynamic demand from known customers (i.e. a priori). The authors refer to this problem and approach combination as the Robust Vehicle Routing Problem (RVRP). Arguably, methods such as sample-scenario as used in [68] could be described as robust control algorithms with soft bounds. In essence they reintroduce probabilities and relax the constraints, in an attempt to choose the most robust solution.

3.3.2.3 Potential Field Control.

Potential field control is a solution that lends itself to real-time control due to its low computational cost [135]. This method generates motion directly from attractive and repellent forces that pull and push the agent respectively. Assigning an attractive force to the goal position and repellent forces to the obstacles can then be used by a hill-climbing solver (or some other local search method) to move the agent. While this method is efficient it suffers from local minima (endemic to all such algorithms) that can trap the agent [111, 158].

Works utilizing this method include [142] which is the problem of collection data from a sparse sensor network. The sensors are given weights with respect to their urgency defined as the amount of data they have that needs to be collected. Other vehicles in the swarm are then given repulsive fields. Similarly, [128] utilizes potential fields to guide Autonomous Underwater Vehicle (AUV) used to create a mobile sensor network. Mei, et al. [108] combined ACO with potential fields to find efficient paths globally and locally respectively. Specifically, the local path planning element repairs the path provided by ACO when obstacles are encountered in unknown and/or dynamic environments. Li and Cassandras [99] present a receding horizon method which utilizes potential fields in a

problem domain that appears to be a variant of the OP (profits assigned to customers with deadlines).

3.3.2.4 Reactive Control.

Reactive control is a method offering low computational costs by reacting to stimuli in a deterministic fashion [135]. For example, if our agent reaches an obstacle, a reactive control rule may direct it to backtrack and alter its original path by a set amount (e.g. backup 1 meter and turn 15°). The problem with such approaches is that the behavior can be difficult to predict in real environments. Such interplay is referred to as emergent behavior [135].

This type of control is arguably a more generalized type of policy as common in solving the DTRP (see Table 3.2 for examples). Alternatively it can be seen as repair operators common to many techniques. For example insertion heuristics in dynamic problems are a reactive method to fix the previous (and now infeasible) solution [154].

3.3.2.5 Reinforcement Learning.

Reinforcement learning is a method which produces a policy via search process. In reinforcement learning, a policy is essentially a deterministic state-to-action lookup. This method requires an accurate model of the domain, a feedback mechanism and a learning period that may be supervised (provided example input-output pairs). One difficulty with this approach is that the agent must tease out which actions were responsible for the reward or penalty which can be difficult in complex problems and/or those with long time frames.

Works utilizing these methods include [117] which applied Q-learning to a problem formulation similar to the DARP. However their formulation used a weighted combination function of customer waiting time and traveling time to form a single objective. It also features vehicles with infinite capacity and, customer locations and reveal times that are not uniform (as in the DTRP). Secomandi [137] utilizes Neuro-Dynamic Programming (NDP) (aka reinforcement learning) to the VRP with dynamic demands for known customers.

Meignan, et al. [109] present a coalition based metaheuristic that has a reinforcement learning component applied to the VRP.

3.3.3 Metaheuristics.

Metaheuristics solve optimization problems as if they were black boxes with simple decisions and an output. This group includes many popular algorithms including Tabu-search, Simulated Annealing, Hill climbing, Ant Colony Optimization (ACO) and Gradient-based methods. Yang characterizes metaheuristics as:

> Most metaheuristics algorithms are nature-inspired ... Two major components of any metaheuristic algorithms are: selection of the best solutions and randomization. The selection of the best ensures that the solutions will converge to the optimality, while the randomization avoids the solutions being trapped at local minima and, at the same time, increase the diversity of the solutions. The good combination of these two components will usually ensure that the global optimality is achievable. [158]

Although metaheuristics are problem domain agnostic, many applications of them include domain-specific heuristics, operators, data types, etc. to improve their performance. This fact blurs the line between heuristics and metaheuristics, but makes their application to problems such as the DMOMCTP easier up front (i.e. specializing a general algorithm is arguably easier than generalizing a specific algorithm).

Metaheristics have been studied extensively in the routing literature, especially in multiobjective formulations due to the difficulty in designing heuristic or exact algorithms. Works using metaheuristics that are similar to the DMOMCTP include Doerner, et al. [49]. That paper compared Parallel Ant Colony Optimization (PACO), Vector Evaluated Genetic Algorithm (VEGA) [136] and Multi-Objective Genetic Algorithm (MOGA) [58] in solving a multiobjective VRP problem with a covering objective for mobile medical facilities.

This study focuses on metaheuristics due to their ease of application to new problems - especially multiobjective problems - and their performance with respect to those problems.

3.4 Summary

MOPs differ from single objective problems in that their solution's utility is a vector. This can be handled using a priori or a posteriori methods but both require a human DM to define their preferences. This process can be difficult and is an active area of research. Since MOP solvers produce sets of solutions, the quality of those sets must be measured which can be difficult. Previous works have produced a number of metrics including hypervolume, generalized spread and generational distance. Number of constraint violations is a measure not confined to MOPs, but for problems like the DMOMCTP are important to consider because finding feasible solutions can be difficult.

The DMOMCTP enjoys a sizable amount of related works, employing a variety of methods for single and multiobjective, static and online, simple and complex routing problems. Not only are routing problems difficult (NP-complete) but the most effective methods can change rather drastically between problem domains. This leaves us with only hints as to what will work well for the DMOMCTP which forms the impetus behind this work.

IV. Methodology

4.1 Native Problem Domain

This experiment simulates a human decision maker (DM) requiring a swarm of Unmanned Aerial Vehicles (UAVs) (i.e. more than two cooperating vehicles) to perform a surveillance mission. The points of interest and travel times between them are provided as a graph for an algorithm to solve centrally (i.e. create a set of routes for the swarm meeting the constraints). The algorithms used are alternatively called solvers. The algorithm provides a set of solutions for the DM to choose from as shown in Figure 4.1. the chosen solution is then sent to and executed by the swarm. A set of solutions is provided (rather than a single ”best” solution) because the DM is not expected to specify their preferences with respect to the multiple objectives a priori. The swarm then *starts* to execute the chosen plan, flying to the nodes in the order specified. During this time, more nodes may be added to the graph by the DM, automated process, etc. (e.g. hostile forces, new surveillance targets). On a regular interval, the central solver resolves the updated problem, providing an updated set of plans to the DM. Again the DM selects a plan from the set which is then distributed to the swarm and executed. This update-plan-select-execute process is repeated until the swarm has completed the mission or it has reached its maximum flight time and must return to the depot.

4.2 Native to Model Problem Domain Mapping

The native problem domain described is modeled as a Dynamic Mulit-Objective Multi-vehicle Covering Tour Problem (DMOMCTP) instance for this experiment. The following subsections cover the model and experiment settings in more detail.

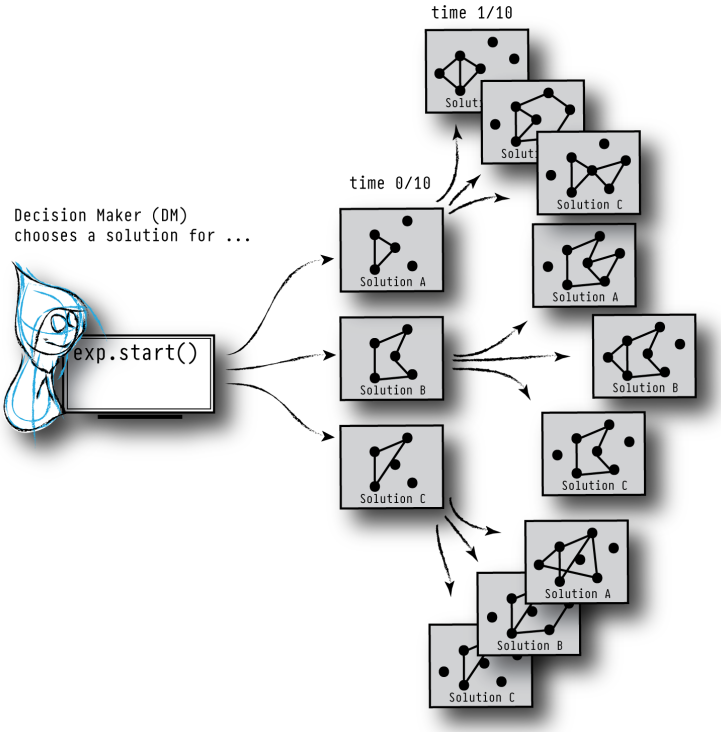


Figure 4.1: Decision Maker Process.

4.2.1 Vehicle Model.

The vehicle parameters used are inspired in part by the General Atomics MQ-1 Predator and MQ-9 Reaper UAVs. The MQ-1's cruising speed is around 84 mph and the MQ-9's is around 230 mph [2, 3]. The MQ-9 vehicle features a Lynx / AN/APY-8 Synthetic Aperture Radar according to Defense Update [4]. The range for this sensor varies from 15.5 to 52 miles depending on weather [5].

Using this information, we chose to model our swarm vehicles as having a speed of 190 mph and covering distance of 75 miles. We chose a speed of 190 mph as a compromise between the MQ-9 and MQ-1 while allowing the vehicle to cover more nodes in our problem instances than would otherwise be possible. The covering distance is increased to 75 miles to compensate for the relatively large distances between nodes in our problem

instances. Using a smaller covering distance would require all nodes to be covered to be visited as no node would exist within the covering distance.

The number of vehicles used is set at four across all of the problems.

4.2.2 Time Model.

The timing parameters include the number of time slices and the solver time limit. To come up with these two parameters, we modeled them on an arbitrary mission which requires replanning once every hour over a ten hour mission. This results in ten time slices.

The planning time limit was set to three minutes. As we are attempting to model an online system with hard-time constraints, this short time limit was felt to be a necessary constraint on the experiment. That said, these parameters are arbitrary and could easily be changed to model different assumptions.

4.2.3 Problem Instances.

The problem instances used are adapted from two Traveling Salesman Problem (TSP) data sets: TSPLIB [7, 132] and Georgia Tech National TSP [6]. From the TSPLIB, we adapted the *usa13509* instance which is based on real Geographic Information System (GIS) data. From the National TSP set, we adapted the *uy734* and *ca4663* instances which are based on real GIS data for Uruguay and Canada, respectively. The GIS data comes from the National Imagery and Mapping Agency database of geographic feature names [6]. These problems are adapted from TSP sets due to their availability, variety of size (number of nodes), ease of use and existing TSP route records. National instances are chosen to better model real-world geographic distributions as opposed to a Random Variable (RV) or other synthetic data set.

To adapt each TSP instance to a DMOMCTP instance, the nodes must be assigned to the various sets along with their reveal times. This is done via Uniform Random Variable (RV). This goes against our desire to use realistic or real-world data when possible, however problems featuring realistic geographic distributions *and* realistic reveal times *and* set

assignment are not available. Since the DMOMCTP is a new problem formulation, even adapting existing Covering Tour Problem (CTP) problem sets would require non-trivial adaptation. Given the trade-offs, we chose to prioritize the realistic geographic distributions over other objectives.

The Random Process (RP) used to assign nodes to sets and reveal times is based on a single Uniform RV. Each node has a 40% chance to be assigned to the must visit set T, 30% chance to be assigned to the cannot visit set O, and a 30% chance to be assigned to the must cover set W. Of those assigned to the must cover set, 17% were also assigned to the cannot visit set O. This creates a relatively small number of nodes which must be covered and avoided, which is meant to model hostile forces. The reveal times of each node is then chosen from the ten time slices via Uniform RV.

4.2.3.1 Dynamism.

The number, rate and distribution of nodes revealed after t_0 are split into two groups: moderately and weakly Effective Degree of Dynamism (EDOD). Instances which are weakly EDOD have 50% of their nodes revealed after t_0 (Degree of Dynamism (DOD) = 0.5) with a Uniform reveal time. Moderately EDOD instances reveal 90% of their nodes after t_0 (DOD = 0.9). Actual EDOD varies depending on the exact number of and time nodes are revealed. For the USA instance, the lightly EDOD problem has an EDOD of 0.22 and the moderately EDOD problem has an EDOD of 0.42. For the Canada instance, the lightly EDOD problem has an EDOD of 0.23 and the moderately EDOD problem has an EDOD of 0.42. For the Uraguay instance, the lightly EDOD problem has an EDOD of 0.23 and the moderately EDOD problem has an EDOD of 0.42.

4.2.3.2 Unit Adaptations.

Treating a unit of distance in the adapted TSP instances as a single mile, the adapted problems would be infeasible within the mission time in the previous section. To accommodate this we use the best known TSP route distance to determine a suitable scaling

factor Q. We also factor in the number of vehicles used and the maximum distance each can travel within the mission time limit. We then use this problem-specific factor to adjust the speed and covering distance of the vehicles. While imperfect, this provided a simple way to scale the problem such that most nodes would be visited by the end of the mission.

First we calculated the unscaled maximum distance covered by all of the vehicles within the mission time: 190 mph $\cdot$ 4 vehicles $\cdot$ 10 hours = 7600 miles/mission. For the USA problem, the best known TSP route provided by the TSPLIB website is 19982859 units (which we treat as miles). Dividing this by our maximum distance and rounding: $Q = \frac{19982859}{7600} = 2629$. Repeating this process for the Canada instance, we get: $Q = \frac{1290319}{7600} = 169$. For the Uruguay instance: $Q = \frac{79114}{7600} = 10$.

Applying these scaling factors to vehicle speed and covering distance, we get $190 \cdot 2629 = 499510$ miles/time slice for the USA problem, $190 \cdot 169 = 32100$ miles/time slice for the Canada problem and $190 \cdot 10 = 1900$ miles/time slice for the Uruguay problem. These speeds are per vehicle.

For covering distance, we used the problem-specific scaling factor multiplied by the covering distance (75 miles). However, in preliminary testing this made the nodes in W too easy to cover. We thus chose to divide this number by 10. Using this equation, results in $\frac{75 \cdot 2629}{10} = 19717$ miles for the USA problem, $\frac{75 \cdot 169}{10} = 1267$ miles for the Canada problem and $\frac{75 \cdot 10}{10} = 75$ miles for the Uruguay problem.

4.2.4 Decision Maker.

For this experiment the DM is an integral factor, making modeling it equally important. However using real human DMs was ruled out due to the time and resources required. This experiment instead uses a Monte Carlo DM which chooses a solution uniformly from the non-dominated solution set produced by the solver at each decision point.

This is a compromise solution, given the many ways it could be modeled. Among its deficiencies are its seemingly arbitrary and changing preferences which we suspect are unlikely to be found in a human DM. Its pros include its ease of implementation and Monte Carlo methods' theoretical properties, namely its consistent and eventual convergence to the true value being estimated given sufficiently large n [84].

Alternative modeling techniques include evaluating each possible non-dominated solution at each decision point, of which there are nine in this experiment. If we assume that each solver produces a maximum of 20 solutions, and all 20 are non-dominated, with nine decision points would give $20^9 = 512,000,000,000$ possible outcomes. Each would, under our assumptions, have 20 solutions in the non-dominated set to evaluate. This is before considering how many algorithms, problems and independent runs we would want to study. This method is not only computationally expensive, it also requires careful design to aggregate those possibilities in a way that correctly represents the *solvers* performance. The major downside of this method is that it still represents the average possible performance of the solver given all preferences. We assume human DMs would instead focus on a smaller range of the objective space.

Another alternative model is to select a random non-dominated solution from the set at the first decision point, but then use some method to choose *similar* solutions thereafter. This could be done by approximating the weights of a weighted combination function given the solution picked. Alternatively weights could be picked first with the best solution according to the resulting combination function chosen at each decision point. Another possibility would be to use a distance metric such that the *closest* solution to the one picked at random was chosen thereafter. These methods are not chosen due to the difficulty in designing good distance metrics, weighted combination functions and/or estimating such weights. In other words, while these methods may more closely model the consistency of human DMs, mathematically defining human DM preferences is a difficult, open problem

in Multiobjective Problem (MOP) research. We chose to use the simpler method in order to focus our efforts on exploring the DMOMCTP.

4.3 Simulation Execution

4.3.1 High-Level Design.

The simulated mission begins by providing the graph as it is currently known at time t_0 to the solver. The solver is given a limited time to run after which it is forced to stop and provide its non-dominated set of plans. The simulated DM then randomly selects plans from the set (see Figure 4.1). The simulator then calculates where the vehicles will be and what nodes they will have visited by t_1 and fast forwards to the beginning of the next replanning phase. The solver is provided the updated graph and is again given a limited amount of time to come up with the best set of plans it can. The resulting non-dominated set is again provided to the DM as in Figure 4.2. The DM again selects a random non-dominated plan to then simulate.

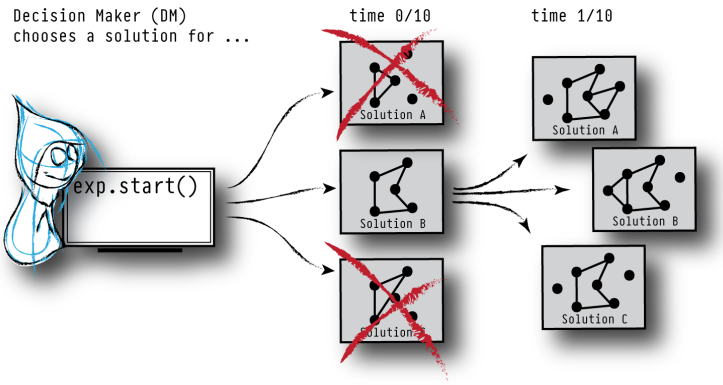


Figure 4.2: Decision Maker Process at t_1.

This cycle continues until the last nodes are revealed at $t_{|M|}$ and the planner provides its set of solutions. Instead of the DM choosing a plan, this final set is saved. The

simulation then continues, iterating through each algorithm and problem combination for each independent run. After performing each assigned run, the simulation ends.

After the simulation has completed, each algorithm's resulting non-dominated set is evaluated against P_{known} for the corresponding problem instance. Note this set of solutions represents possible future actions, but past actions taken are taken into account when calculating their utility.

The experiment uses 50 independent runs of each algorithm for each problem instance to approximate the efficacy of each algorithm. This is required to better approximate the solver's efficacy since the DM and each solver is stochastic. More independent run would be desirable, however doing so would require significantly more computation.

4.3.2 Implementation.

The experiment utilizes the algorithm and metric code from jMetal 4.2 with the rest of that framework being replaced. The codebase differs from jMetal 4.2 in that it can handle the time and DM dependent nature of the DMOMCTP. The algorithms and new code can handle variable length solutions, additional operators, and additional algorithms. The new framework persists results to a remote CouchDB database allowing multiple instances of the experiment to run over several identical computers while easily aggregating the results for later analysis as shown in Figure 4.3. Each independent run encompasses a single run of each algorithm over each problem instance.

The experiment was run on Amazon's EC2 using 51 m1.small instances. These are virtualized servers with 1.7 GiB memory, 1 EC2 compute unit (1 virtual core with 1 EC2 compute unit) and 160 GB storage [1]. Amazon defines 1 EC2 compute unit as "provid[ing] the equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor."[1] Fifty of the instances are used to run the experiment with one used to host the CouchDB server. While the exact type of processor is unknown, the experiment has been designed to study the *relative* performance of each solver given the same time limit. As each m1.small

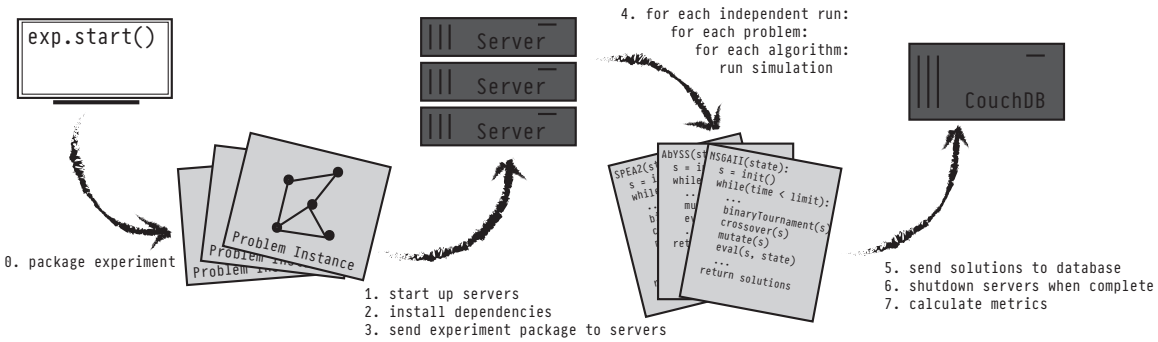


Figure 4.3: Experiment Execution.

instance is guaranteed to have the same amount of processing power and each server runs each algorithm for each problem, this assumption holds and each run's results should be comparable.

4.3.3 Implementation Limitations.

Due to limitations in Java's Double type and the size of the problems studied, objectives may reach Double.Infinity. This complicates comparisons and metrics used in the experiment. We chose to convert these rare instances to Double.MAX_VAL results in a slight distortionary effect for the results.

4.4 Performance Measures

The metrics used for comparing these algorithms are hypervolume, generalized spread, generational distance and number of constraints violated. Except for number of constraints violated, all are presented as a percentage of P_{known}. These metrics are presented in relative terms to aid understanding. The set P_{known} is created by aggregating all non-dominated solutions created by all of the algorithms and independent runs.

4.5 Solvers Studied

Several solvers are included in the experiment, along with several different parameters and operators. The experiment is not a full factorial design in part due to the number of unique combinations and amount of computation required to do so. For this experiment we use six problems, ten algorithms, ten time slices and a three minute time limit for each run. Preliminary results and past works are used to choose which algorithm permutations to include.

The algorithms studied include Archive-based Scatter Search (AbYSS) [119], Nearest Neighbor with Random Weights (NNRW), Nondominated Sorted Genetic Algorithm II (NSGA-II) [46], Strength Pareto Evolutionary Algorithm 2 (SPEA2) [160], and Random Search (RS). All algorithm code except for NNRW and RS is from jMetal version 4.2. AbYSS, NSGA-II, and SPEA2 are chosen due to their past performance record with respect to a posteriori MOP optimization and their metaheuristic nature which makes them easily adaptable to the DMOMCTP. RS is included for baseline comparison to a random walk. NNRW is included due to its simplicity, and the Nearest Neighbor (NN) algorithm's performance with respect to Dynamic Traveling Repairperson Problem (DTRP). It also represents an algorithm which would be straightforward to decentralize.

AbYSS [119] is based on the common scatter search template, but includes ideas from other Multiobjective Evolutionary Algorithm (MOEA). It utilizes the archive curation method from Pareto Archived Evolution Strategy (PAES), the crowding distance mechanism from NSGA-II and the density measure from SPEA2 to select solutions for the reference set.

NNRW is a modified version of the greedy, deterministic Nearest Neighbor algorithm common in DTRP works. The algorithm builds a solution by first randomly choosing weights for the combination function shown in Equation (4.1). Each weight is chosen via Uniform RV over the range 0.0 to 1.0. This function provides NNRW with a single

objective to minimize. It then iteratively chooses the cheapest node assignment according to this function from the set of visitable nodes. After each assignment, it keeps the solution if no other solution dominates it. If kept, it then removes solutions dominated by the newly added solution. This process continues until 60% of the candidate node set has been used. NNRW then starts a new solution (randomly choosing the new weights and iteratively building a solution).

$$\text{utility}(s) = af_1(s)^{aa} + bf_2(s)^{bb} + cf_3(s)^{cc} + df_4(s)^{dd} \tag{4.1}$$

NSGA-II [46] is based on the original Nondominated Sorted Genetic Algorithm (NSGA) and is common in MOEA comparisons. This version of NSGA ranks each population member according to non-domination level then applies selection, crossover, and mutation operators. It uses niching via a crowding distance metric that it assigns to each population member to maintain a diverse population.

RS creates population members randomly from the visitable set of nodes, maintaining those which are non-dominated. The population itself is unbounded and no other operators are used except for an optional repair chain. This algorithm is included as a baseline for comparison.

SPEA2 [160] builds upon the original Strength Pareto Evolutionary Algorithm (SPEA) by crafting a more detailed fitness measurement of each population member. It does this by factoring in the number of population members which dominate it and the number it dominates. It also uses a nearest neighbor density measurement and a different archive management function that maintains key boundary population members.

4.5.1 Solution Types.

The experiment uses two data types for solutions: fixed and variable length. The use of fixed length solutions offers potentially simpler operator design and faster manipulation. Variable length solutions are more intuitive for problems such as the DMOMCTP which

can fluctuate in size and where a fixed-length version would often include unused decision variables requiring more computation to operate on. Each algorithm is used with both solution types when possible. Table 4.1 lists solution type incompatibilities.

Fixed length solutions in this experiment are actually masked variable length solutions. They use the same underlying LinkedList data type to store the routes, but initialize them to the desired size and provide operators for direct manipulation. This results in O(1) reads and writes which is the main desirable facet of fixed length solutions (along with identical sized solutions for certain operators).

Table 4.1: Algorithm, Solution Type and Operator Incompatibilities.

Type	Operator / Setting	Var-Len	Fix-Len	RS	NSGAII	AbYSS	SPEA2
	Variable Length Solution			✗		✗	
	Seed Population			✗			
Initializer	Uniform (visitable nodes)						
	Uniform (all nodes)						
Selection	Binary Tournament						
	Binary Tournament 2						
Crossover	SBX	✗		✗			
	Route Swap (Uniform 0.30)			✗			
Mutation	Polynomial						
	Mutation Local Search			✗	✗		✗
Deletion	Uniform (0.05)		✗	✗		✗	
Insertion	Addition (Uniform 0.06)		✗	✗		✗	
	Duplication (Uniform 0.06)		✗	✗		✗	
Repair	Monolithic remove consecutive duplicate visits						
	Monolithic remove all duplicate visits						
	Monolithic remove undiscovered nodes						
	Monolithic remove visiting constraint violations						
	Monolithic fill with must visit nodes						
	Monolithic fill with visitable nodes						

✗ denotes an incompatibility between the operator, solution type and/or algorithm

4.5.2 Operators.

The choice and order of operators is an important part of MOEA design. For this experiment, several different operators are included both from the base algorithm and from the literature.

4.5.2.1 Initialization.

Initialization operators create the initial population members. A single initializer is used in this work that can create both variable and fixed length solutions from a pool of candidate values. These candidate values are initialized to be the set of currently visitable nodes. For fixed length solutions, it creates k routes with a fixed length. Each variable in each route is then chosen from the candidate values with a uniform distribution. For variable length solutions, it first chooses how many nodes to insert using a Gaussian distribution ($\mu = 0.3 * \text{num of candidates}$; $\sigma = 0.25 * \text{num of candidates}$) and then chooses which nodes to insert from the set of candidates (Uniform RV).

4.5.2.2 Selection.

The experiment utilizes two different versions of the same type of selection: binary tournament (referred from here on as Binary Tournament 1 and Binary Tournament 2). Both versions randomly choose two members of the population to compare. If one dominates the other according to the chosen comparator, it is chosen as the winner. Both versions use a comparator that first tests for domination by number of constraints violated (less is better) and if no difference exists by that measure, by Pareto dominance. The two versions differ in how they handle the case where neither solution dominates the other. Binary Tournament 2, uses a crowding distance metric to decide, returning whichever has the greater crowding distance (creating a more diverse population). Binary Tournament 1 skips this crowding comparison. Both versions of binary tournament choose a winner randomly if none of the measures above provide an answer.

4.5.2.3 Crossover.

Simulated Binary Crossover (SBX) is used on real-valued solution representations. It attempts to simulate the effects of a single point crossover operator on a binary-encoded solution.

Route Based Crossover (RBX) is a crossover operator designed for solutions that encode multiple routes such as in the DMOMCTP. It iterates through the parents one route at a time, swapping the current route given a certain probability (using a Uniform RV). For example, assume solution A has two routes (2, 4) and (3, 1) and solution B has two routes (2, 1) and (3, 5, 6). RBX lines them up side by side and starts at route one, swapping A and B's first routes with a probability p. If RBX swapped route one and no other route, solution A would become (2, 1), (3, 1) and solution B would become (2, 4), (3, 5, 6).

4.5.2.4 Mutation.

Polynomial mutation iterates through all of the solution member's variables, altering each with a probability of $\frac{1}{numvars}$. When it does alter a single variable, it takes that variable's minimum, maximum and current value to create a distribution to sample, with the current value as its mean.

For variable length solutions, additional mutation operators are included which can be grouped into insertion and deletion sub-types. Insertion operators add one or more variables to the solution member, whereas deletion removes one or more. One insertion operator - referred to here simply as addition - adds a single variable with a probability p choosing its location and value uniformly (choosing from the set of currently visitable nodes). Duplication is another insertion operator which, with a probability p chooses an existing variable from the solution (using a Uniform RV) and duplicates it right beside the original. The only deletion operator tested deletes a single variable from a solution with a probability p (choosing the variable via Uniform RV).

4.5.2.5 Repair.

A single repair operator is used in this experiment, referred to as the monolithic repair operator. This operator attempts to correct solutions which violate constraints and utilizes domain knowledge to nudge solutions in promising directions right before evaluating a

solution. Thus All of its functionality can be turned on or off via flags, some overriding others.

Table 4.2: Monolithic Repair Operator Flags.

Flag	Description
r_1	Remove consecutive duplicate variables
r_2	Remove all duplicate variables
r_3	Remove undiscovered nodes
r_4	Remove visiting constraint violations
r_5	Fill with visitable nodes
r_6	Fill with must-visit nodes

Removing consecutive duplicate variables (r_1) removes any variable that is repeated except for the depot. For fixed length solutions, this essentially shifts all variables down one, with the missing variable at the end being reinitialized via r_5, r_6 or being set to the depot depending on the flags. The *remove all duplicate variables* flag (r_2) removes all duplicates, even those appearing in different routes or those separated by one or more variables. This flag overrides r_1 if set. The empty variables created (for fixed length solutions) are filled in the same fashion as those from r_1.

Removing undiscovered nodes (r_3) removes nodes that are used but are not known to exist in the current state. This can occur during mutation (and other operators) as not all operators use domain knowledge to determine valid values outside of the range allowed (e.g. polynomial mutation). *Removing visiting constraint violations* (r_4) builds on this notion by also including nodes which are known in the current time slice, but which cannot be visited (i.e. are in the set O). This flag overrides r_3 if set. If a node is removed, all others shift down and is filled in the same fashion as those from r_1.

If a fixed length solution is used and a variable is removed (via one of the repairs) the rest of the nodes are shifted down and the empty variable is initialized to the depot *by default*. Since all routes in the DMOMCTP must begin and end at the depot, this has the

effect of creating a no operation (no-op). This is why the depot is ignored when considering duplicates. However, this behavior is changed if either r_5 or r_6 is set. The *fill with visitable nodes* flag (r_5) uses domain knowledge to choose a random visitable node which is then inserted into a random route at a random location. The inserted value is removed from the set of candidates (visitable nodes). The operator stops either at the end of each insertion (25% chance) or when it runs out of candidate node set. If the *fill with must visit nodes* flag (r_6) is set, the candidate nodes is instead set to those which must be visited (r_6 overrides r_5), and there is no chance to stop at the end of each insertion (i.e. it stops only when all of the must visit nodes have been inserted).

Both of the previously mentioned insertion repairs (r_5 and r_6) can add too many variables in the case of fixed length solutions. To accommodate for this, the monolithic repair operator always checks the fixed-length solutions for length and removes variables as needed. This apparent contradiction of terms (fixed length solutions which are too long) is due to the underlying implementation of fixed length solutions.

Monolithic repair then completes by verifying each route ends at the depot. If not, it overwrites the last variable for fixed length solutions or adds it for variable length solutions.

Based on preliminary testing, we use either no repair operator or a monolithic repair with the r_2, r_4, and r_6 flags turned on. Those flags remove all duplicate visits, remove visiting constraint violations and fill in fixed-length solutions with must-visit nodes. This offers a perspective on the effects of a strong repair operator and without one.

4.5.3 Parameters.

Each algorithm and solution type requires several parameters to be set. Most are left at the defaults set in the original jMetal 4.2 code. Those that are modified were only altered after preliminary tests showed them to be ineffective. For example, NSGA-II and SPEA2 both have their population parameter lowered to 20. Preliminary testing found that using the default (100) resulted in few if any iterations over the entire population and poor non-

dominated solution sets with respect to the three minute time limit. Table 4.3 lists the parameters used in the algorithms and operators.

Table 4.3: Algorithm Paramaters.

Algorithm	Paramater	Value	Operator Parameters
AbYSS	Solution Set Size	20	
	Archive Size	100	
	Reference Set 1&2 Size	10	
	Number of Subranges	4	
	Crossover	SBX	Probability: 1.0 Distribution Index: 20
	Improvement	Mutation Local Search	Improvement Rounds: 1 Mutation: Polynomial
NSGA-II	Population Size	20	
	Mutation	Polynomial Mutation	Distribution Index: 20
	Crossover (fix-len)	SBX	Probability: 1.0 Distribution Index: 20
	Crossover (var-len)	RBX	Probability: 0.3
	Selection	Binary Tournament 2	includes niching via crowding distance
	Deletion	Uniform Deletion	Probability: 0.05
	Insertion	Duplication	Probability: 0.06
SPEA2	Population Size	20	
	Archive Size	100	
	Crossover	SBX	Probability: 0.9 Distribution Index: 20
	Mutation	Polynomial Mutation	Distribution Index: 20
	Selection	Binary Tourament 1	no niching / crowding distance
	Deletion	Uniform Deletion	Probability: 0.05
	Insertion	Duplication	Probability: 0.06

4.5.4 Algorithms Tested.

From the options we test ten algorithms, including two versions of AbYSS and three versions of both NSGA-II and SPEA2. The complete list is provided in Table 4.4. Those included are selected due to their performance in preliminary testing and past MOP works, but are ultimately arbitrary without evidence with respect to the DMOMCTP.

RS is included to provide a baseline comparison. NNRW is included to explore a possibly simple decentralized solution. AbYSS does not include a variable length variation because it is incompatible with the variable length solution type SPEA2 and NSGA-II

Table 4.4: Algorithm Combinations Tested.

Short Name	Solution Type	Crossover	Insertion	Deletion	Repair
RS 0	Variable	-	-	-	None
NNRW 1	Variable	-	-	-	None
AbYSS 2	Fixed	-	-	-	None
AbYSS 3	Fixed	-	-	-	Full
SPEA2 4	Fixed	SBX	-	-	Full
SPEA2 5	Variable	RBX	Unif. Addition	Unif. Deletion	None
SPEA2 6	Variable	RBX	Unif. Addition	Unif. Deletion	Full
NSGAII 7	Fixed	SBX	-	-	Full
NSGAII 8	Variable	RBX	Unif. Addition	Unif. Deletion	None
NSGAII 9	Variable	RBX	Unif. Addition	Unif. Deletion	Full

both include one fixed length version and two variable length versions due to the poor performance we found in preliminary testing of the fixed length solution type.

Most algorithms include a version that uses the monolithic repair and one that does not. The repair operator used - referred to as "full-repair" from here on - set the r_2, r_4 and r_6 flags. These flags remove all duplicate visits, remove visiting constraint violations and fill in fixed-length solutions with must-visit nodes. This offers a perspective on the effects of a strong repair operator. The full-repair is not used with RS or NNRW to reduce the amount of computational time the experiment requires.

For the algorithm combinations that use variable length solution types - SPEA2 and NSGA-II - we use the RBX for the crossover operator, Uniform addition for the insertion operator and Uniform deletion for the deletion operator. We found this mix to work the best in preliminary testing.

Code for AbYSS, SPEA2 and NSGA-II comes from jMetal 4.2, and each use the default parameters. The only exception is that SPEA2 and NSGA-II use a population of 20. As previously discussed this was as result of poor performance in preliminary testing which we believe is due to the low number of iterations they were able to complete over the population given the time limit.

4.6 Summary

This chapter briefly covered the native problem domain and mapped it to our DMOMCTP model. The human DM is modeled as a Uniform RV despite that method's shortcomings. For problem instances, modified several TSP problems are used originating from GIS data to better model the geographical distributions of nodes. The vehicle settings used arbitrary numbers but took into account public details for the MQ-1 and MQ-9 UAVs. For time, ten time slices, with a three minute time limit is used. Hypervolume, generalized spread, generational distance and number of constraints violated are used for measuring the solver's performance.

To avoid the large computational time required for a full-factorial design the study compares ten algorithm combinations. Two versions of AbYSS, and three for both SPEA2 and NSGA-II, with RS for a baseline and NNRW for a naive, but easily decentralized method.

To better approximate the solver's performance, each algorithm is run over each problem fifty times. To reduce the wall-clock time required, each independent run is distributed across fifty servers. When completed, the results are then fetched from CouchDB and the metrics calculated.

V. Results

This section reviews the experimental results and analysis. Note that the use of multiple metrics for measuring performance of a posteriori Multiobjective Problems (MOPs) solver makes analysis a Multiobjective Problem (MOP) itself. In other words, determining which algorithm is best requires a human decision maker (DM) to define their preferences [38]. For example, one DM may prefer a method which produces solutions with more feasible solutions over one that produces a better spread or hypervolume.

This chapter discusses the relationships between methods, problems and metrics. It also provides a rough ordering for methods when one method shows itself to dominate the other.

Section 5.1 presents the data via violin and three dimensional solution set plots for each problem instance. Section 5.3 provides an algorithm-level analysis of the data and Section 5.4 provides an operator-level analysis. Section 5.5 reviews the analysis and high-level conclusions.

5.1 Data Presentation

Figure 5.1 through Figure 5.11 provide violin plots for number of constraints violated, hypervolume and generalized spread. Violin plots are a combination of box plot and kernel density plot. Number of constraints violated and generalized spread are to be minimized while hypervolume is to be maximized. Additional measures including generational distance are provided in Appendix A, but are left out in Figure 5.1-Figure 5.11 due to the low amount of variance found by those measures. Epsilon shows variance among the algorithms, but is quite similar to the results found by hypervolume. Appendix A also includes a larger format of the three metrics this section uses.

Table 4.4 is reproduced below each set of metric plots for reference. Each algorithm is given a short name which is its base algorithm acronym and a number. The number portion is unique among all algorithms for easy identification.

Each figure is one problem instance, with one for light and moderate dynamism as measured by Effective Degree of Dynamism (EDOD). Hypervolume and generalized spread are on a scale of 0.0 to 1.0 which represents the ratio of that algorithm's metric with respect to P_{known} as discussed in Section 3.1.4.

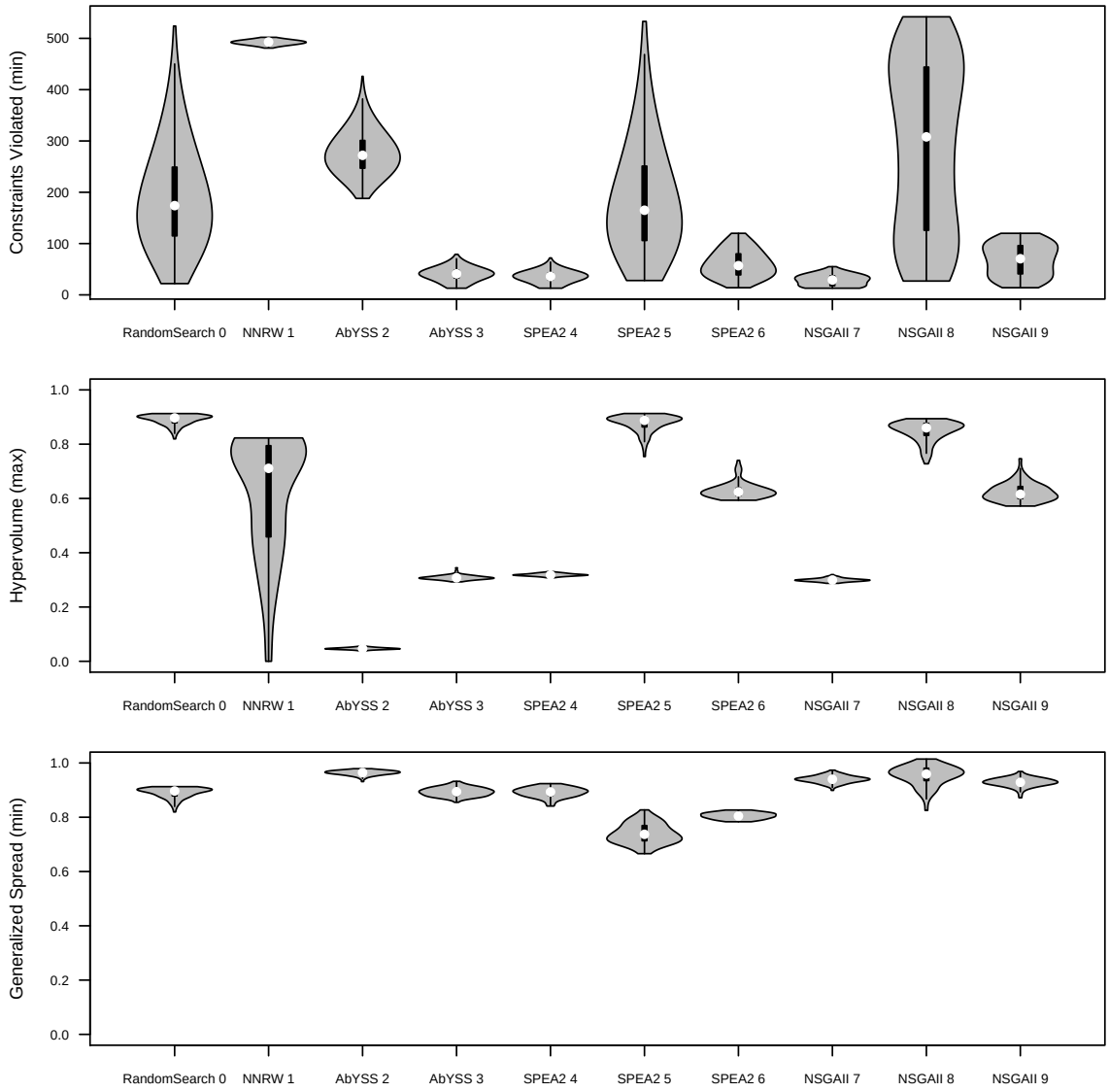
After the violin plots, three dimensional plots are provided of all solutions sets returned by each solver. The first, second and third objective are assigned to the x, y, and z axis respectively while the fourth objective is shown via color. This is done because f_1, f_2, and f_3 are to be minimized while f_4 is to be maximized. Better solutions are those closer to the plots' origin and closer in color to yellow than to red. Note it does not depict the number of constraints violated.

The first column of the solution set plots includes all of the algorithms which use fixed-length solution encoding, some of which use the repair and others do not. The second column includes all of the solvers which use variable-length solution encoding and no repair while the last column uses variable-length solutions and the monolithic repair. Appendix B provides a larger format of all the solution set plots.

While plotting the fronts for higher-dimensional problems can be misleading [38] they are provided to supplement the analysis because they provide a useful view of the problems and algorithms.

5.2 Data

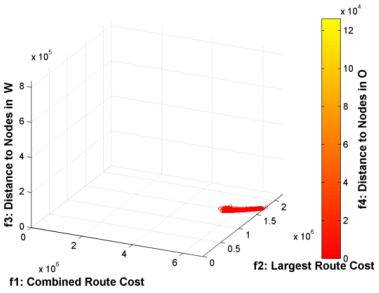
Figure 5.1: Uruguay 734 - Light EDOD - Violin Plots



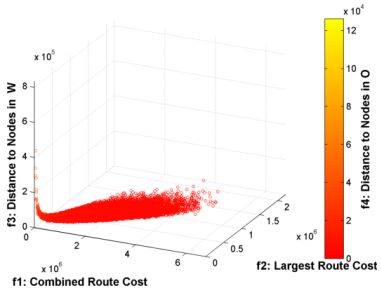
Short Name	Solution Type	Crossover	Insertion	Deletion	Repair
RS 0	Variable	-	-	-	None
NNRW 1	Variable	-	-	-	None
AbYSS 2	Fixed	-	-	-	None
AbYSS 3	Fixed	-	-	-	Full
SPEA2 4	Fixed	SBX	-	-	Full
SPEA2 5	Variable	RBX	Unif. Addition	Unif. Deletion	None
SPEA2 6	Variable	RBX	Unif. Addition	Unif. Deletion	Full
NSGAII 7	Fixed	SBX	-	-	Full
NSGAII 8	Variable	RBX	Unif. Addition	Unif. Deletion	None
NSGAII 9	Variable	RBX	Unif. Addition	Unif. Deletion	Full

Figure 5.2: Uruguay 734 - Light EDOD - Pareto Fronts

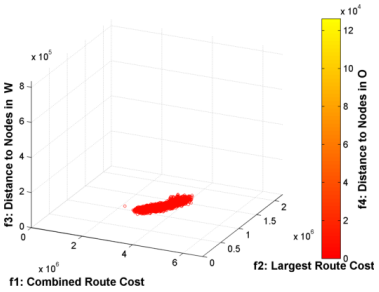
AbYSS 2



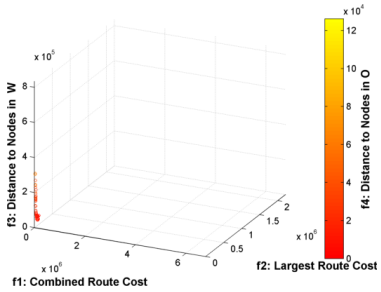
Random Search 0



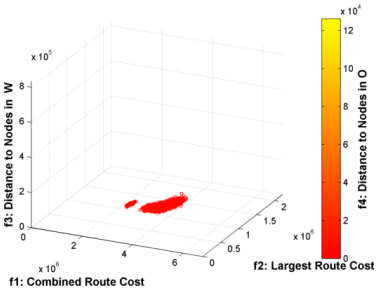
AbYSS 3



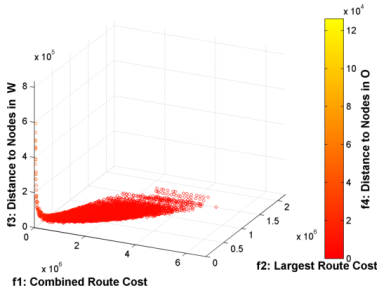
NNRW 1



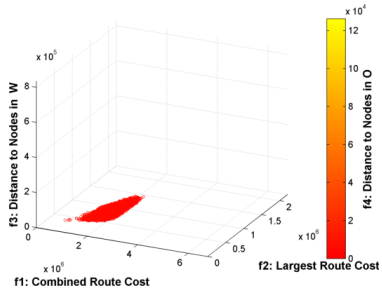
SPEA2 4



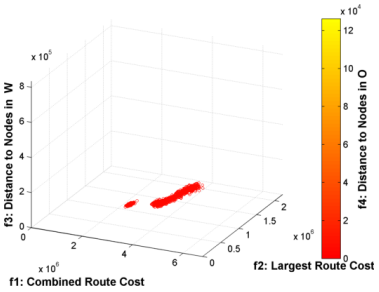
SPEA2 5



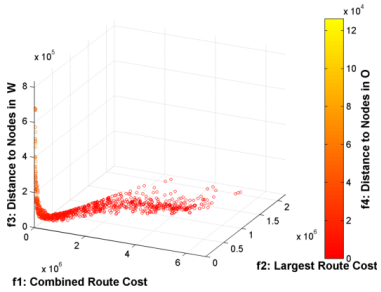
SPEA2 6



NSGA-II 7



NSGA-II 8



NSGA-II 9

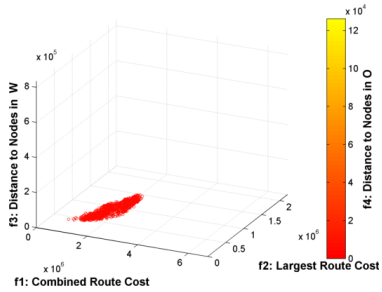
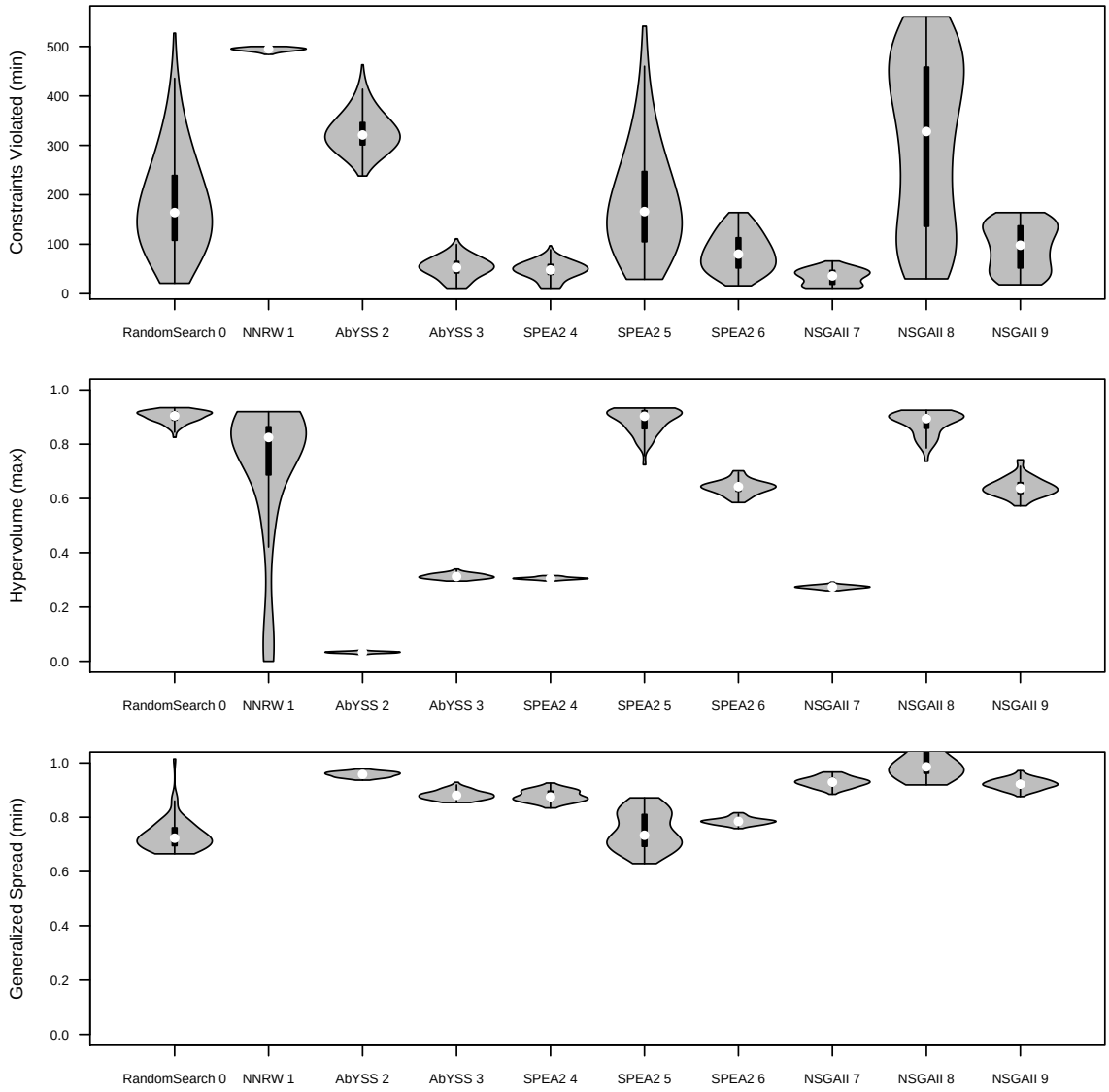


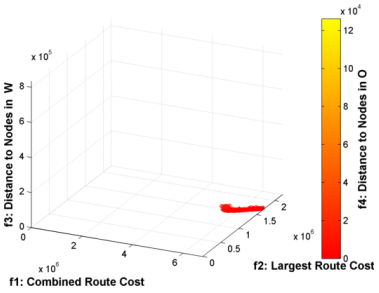
Figure 5.3: Uruguay 734 - Moderate EDOD - Violin Plots



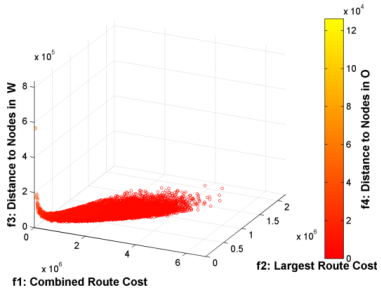
Short Name	Solution Type	Crossover	Insertion	Deletion	Repair
RS 0	Variable	-	-	-	None
NNRW 1	Variable	-	-	-	None
AbYSS 2	Fixed	-	-	-	None
AbYSS 3	Fixed	-	-	-	Full
SPEA2 4	Fixed	SBX	-	-	Full
SPEA2 5	Variable	RBX	Unif. Addition	Unif. Deletion	None
SPEA2 6	Variable	RBX	Unif. Addition	Unif. Deletion	Full
NSGAII 7	Fixed	SBX	-	-	Full
NSGAII 8	Variable	RBX	Unif. Addition	Unif. Deletion	None
NSGAII 9	Variable	RBX	Unif. Addition	Unif. Deletion	Full

Figure 5.4: Uruguay 734 - Moderate EDOD - Pareto Fronts

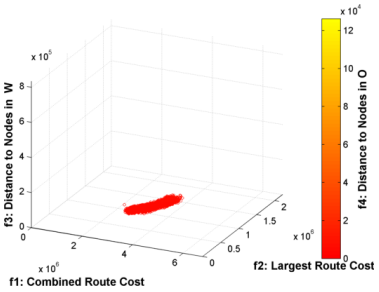
AbYSS 2



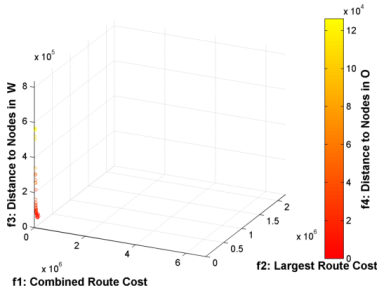
Random Search 0



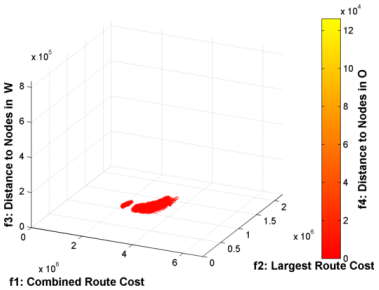
AbYSS 3



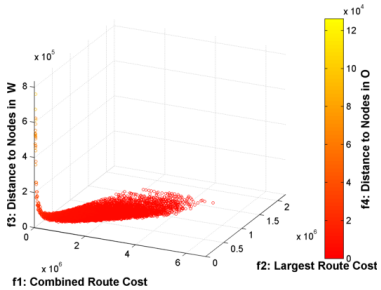
NNRW 1



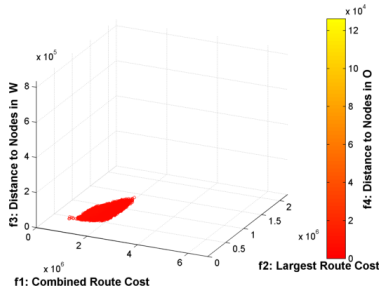
SPEA2 4



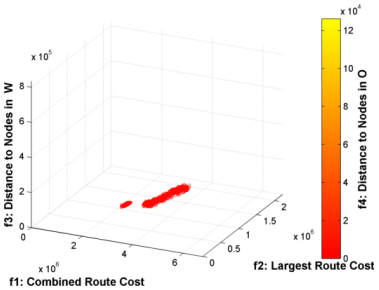
SPEA2 5



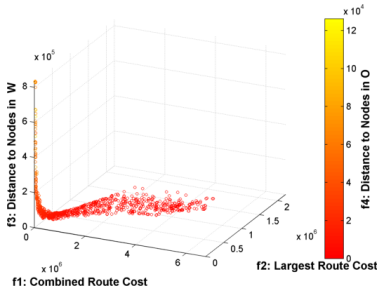
SPEA2 6



NSGA-II 7



NSGA-II 8



NSGA-II 9

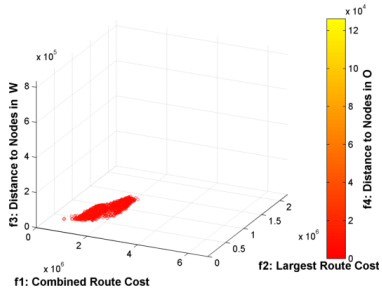
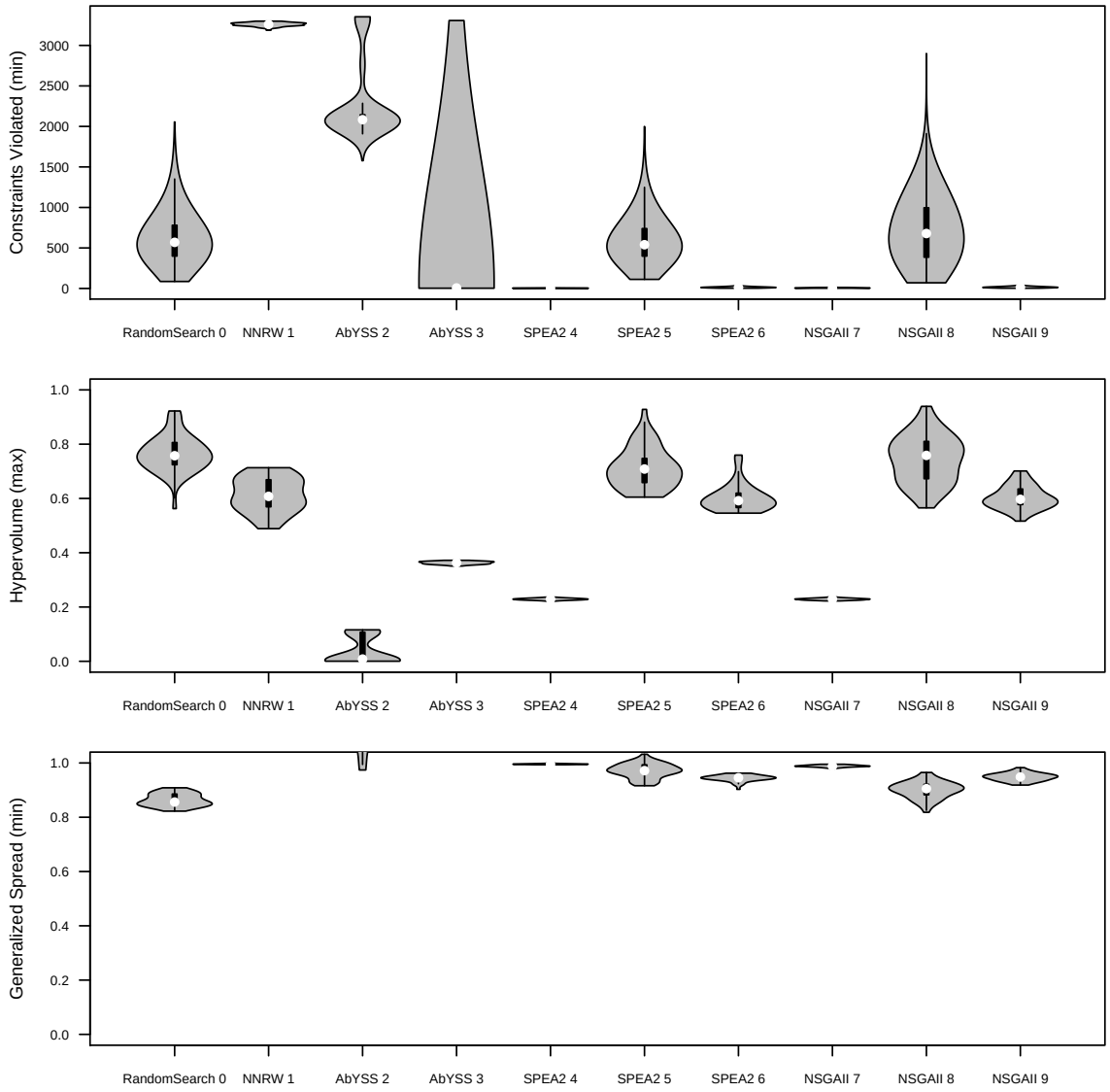


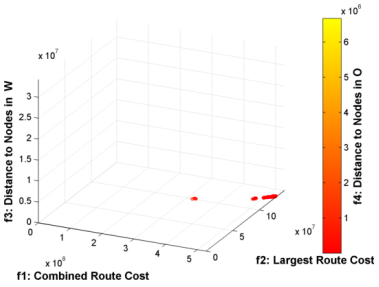
Figure 5.5: Canada 4663 - Light EDOD - Violin Plots



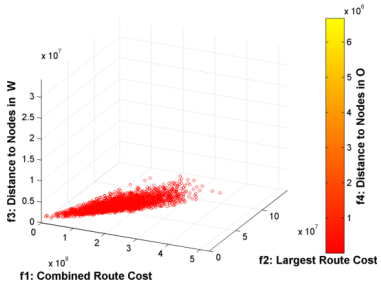
Short Name	Solution Type	Crossover	Insertion	Deletion	Repair
RS 0	Variable	-	-	-	None
NNRW 1	Variable	-	-	-	None
AbYSS 2	Fixed	-	-	-	None
AbYSS 3	Fixed	-	-	-	Full
SPEA2 4	Fixed	SBX	-	-	Full
SPEA2 5	Variable	RBX	Unif. Addition	Unif. Deletion	None
SPEA2 6	Variable	RBX	Unif. Addition	Unif. Deletion	Full
NSGAII 7	Fixed	SBX	-	-	Full
NSGAII 8	Variable	RBX	Unif. Addition	Unif. Deletion	None
NSGAII 9	Variable	RBX	Unif. Addition	Unif. Deletion	Full

Figure 5.6: Canada 4663 - Light EDOD - Pareto Fronts

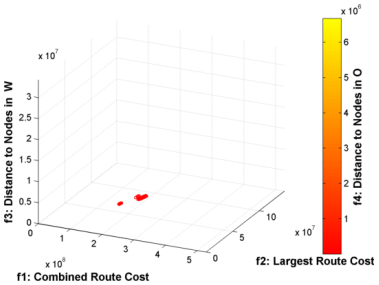
AbYSS 2



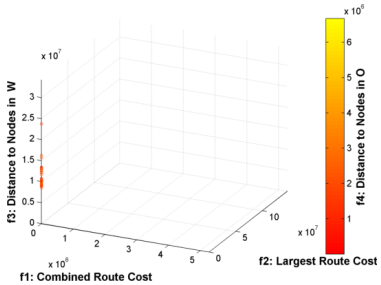
Random Search 0



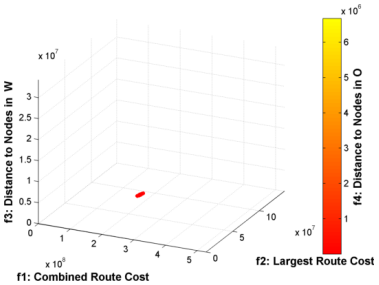
AbYSS 3



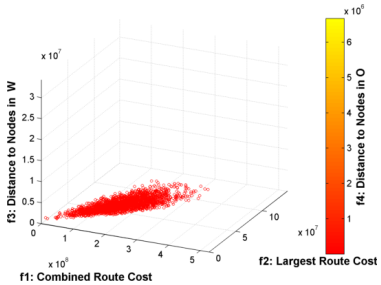
NNRW 1



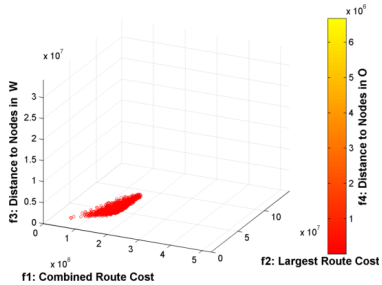
SPEA2 4



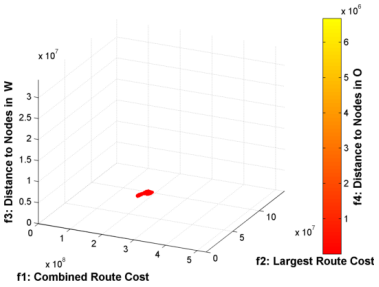
SPEA2 5



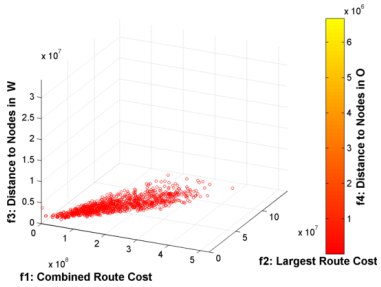
SPEA2 6



NSGA-II 7



NSGA-II 8



NSGA-II 9

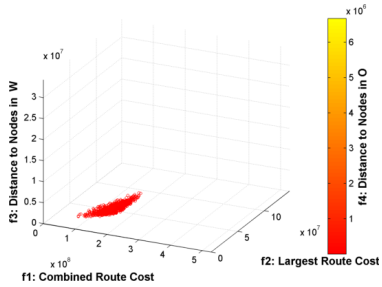
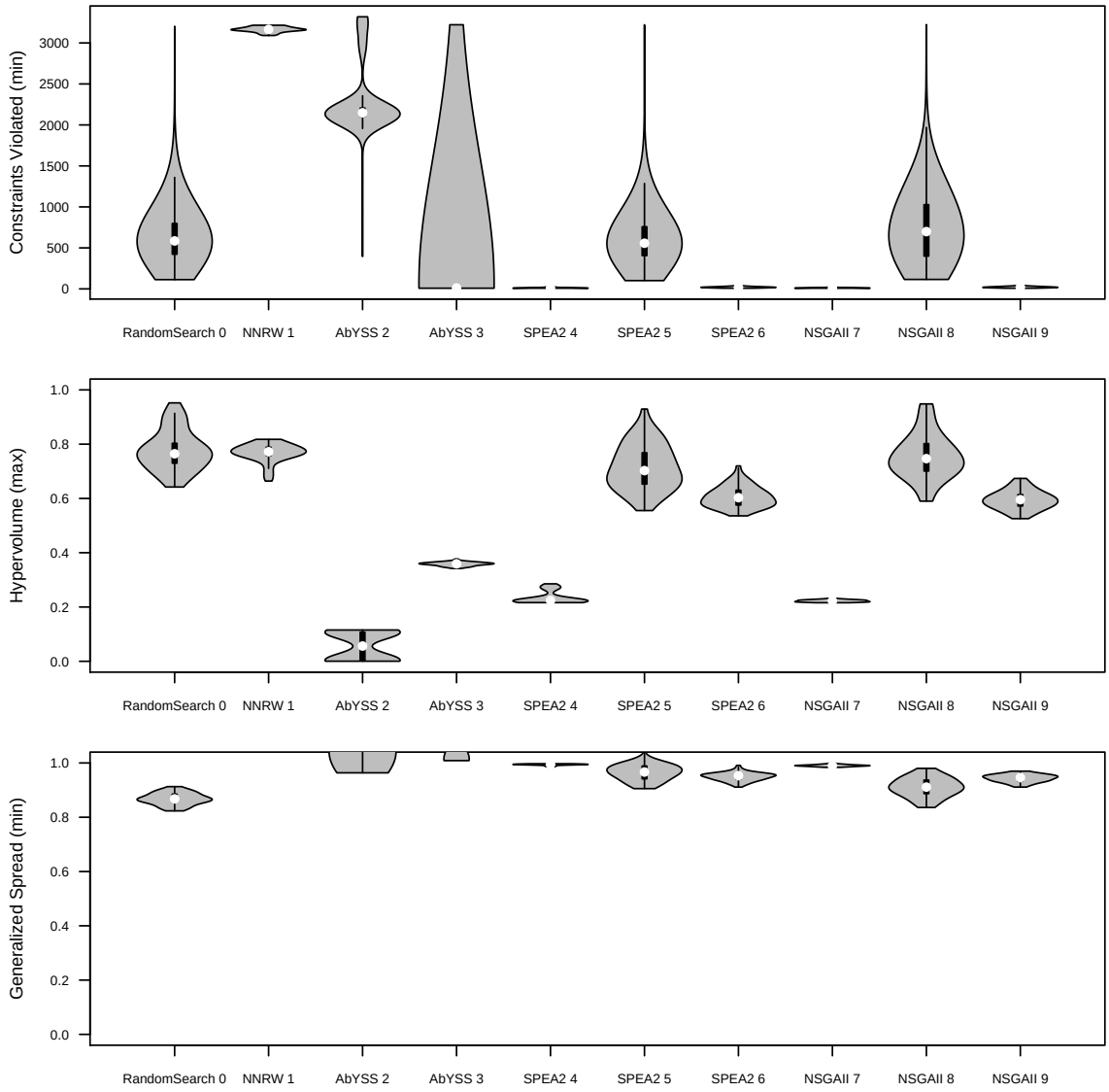


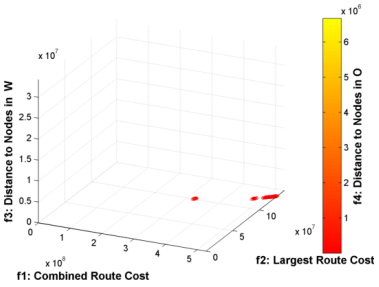
Figure 5.7: Canada 4663 - Moderate EDOD - Violin Plots



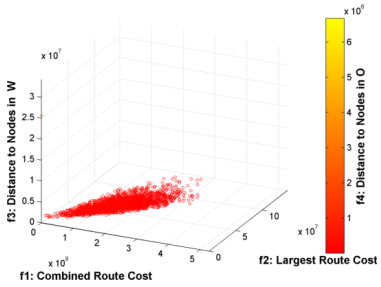
Short Name	Solution Type	Crossover	Insertion	Deletion	Repair
RS 0	Variable	-	-	-	None
NNRW 1	Variable	-	-	-	None
AbYSS 2	Fixed	-	-	-	None
AbYSS 3	Fixed	-	-	-	Full
SPEA2 4	Fixed	SBX	-	-	Full
SPEA2 5	Variable	RBX	Unif. Addition	Unif. Deletion	None
SPEA2 6	Variable	RBX	Unif. Addition	Unif. Deletion	Full
NSGAII 7	Fixed	SBX	-	-	Full
NSGAII 8	Variable	RBX	Unif. Addition	Unif. Deletion	None
NSGAII 9	Variable	RBX	Unif. Addition	Unif. Deletion	Full

Figure 5.8: Canada 4663 - Moderate EDOD - Pareto Fronts

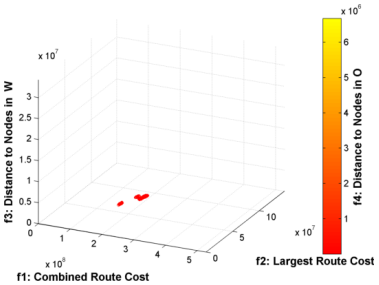
AbYSS 2



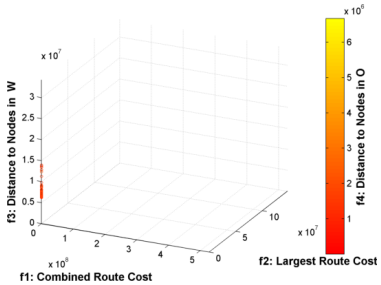
Random Search 0



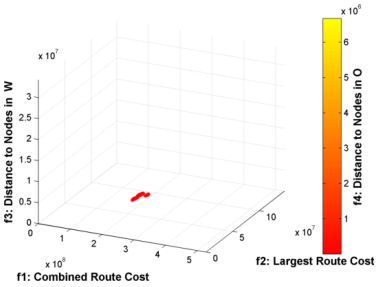
AbYSS 3



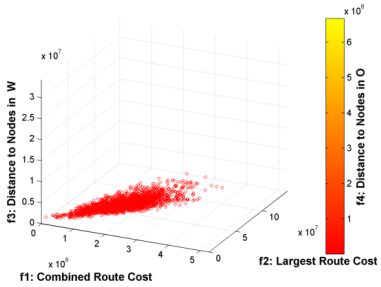
NNRW 1



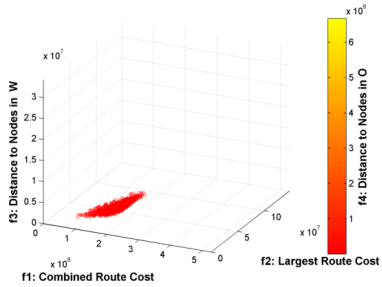
SPEA2 4



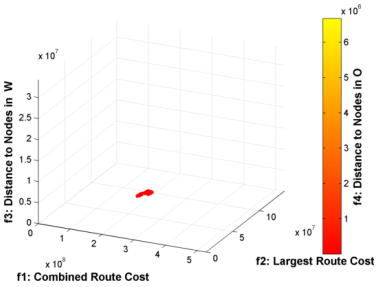
SPEA2 5



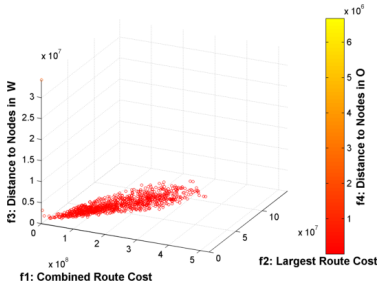
SPEA2 6



NSGA-II 7



NSGA-II 8



NSGA-II 9

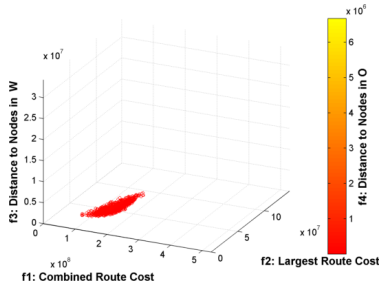
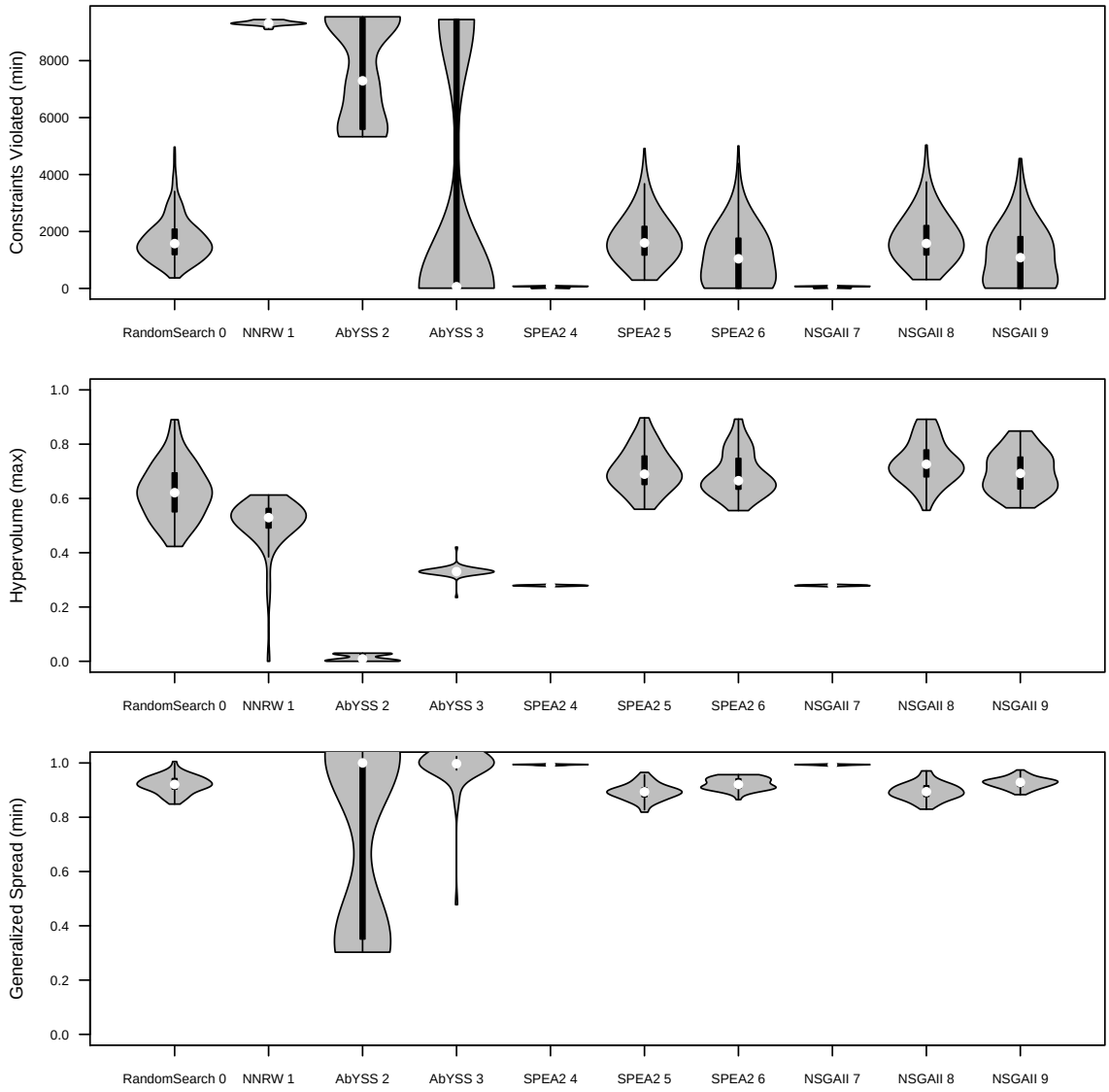


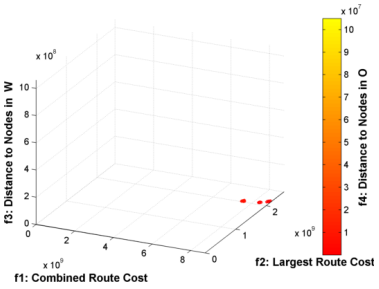
Figure 5.9: USA 13509 - Light EDOD - Violin Plots



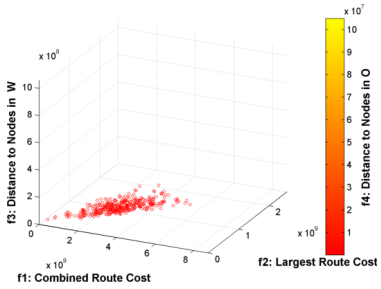
Short Name	Solution Type	Crossover	Insertion	Deletion	Repair
RS 0	Variable	-	-	-	None
NNRW 1	Variable	-	-	-	None
AbYSS 2	Fixed	-	-	-	None
AbYSS 3	Fixed	-	-	-	Full
SPEA2 4	Fixed	SBX	-	-	Full
SPEA2 5	Variable	RBX	Unif. Addition	Unif. Deletion	None
SPEA2 6	Variable	RBX	Unif. Addition	Unif. Deletion	Full
NSGAII 7	Fixed	SBX	-	-	Full
NSGAII 8	Variable	RBX	Unif. Addition	Unif. Deletion	None
NSGAII 9	Variable	RBX	Unif. Addition	Unif. Deletion	Full

Figure 5.10: USA 13509 - Light EDOD - Pareto Fronts

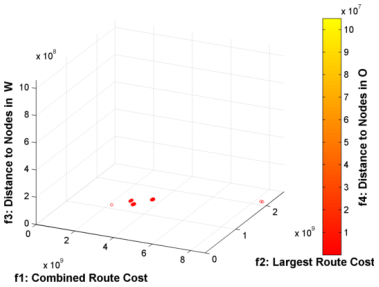
AbYSS 2



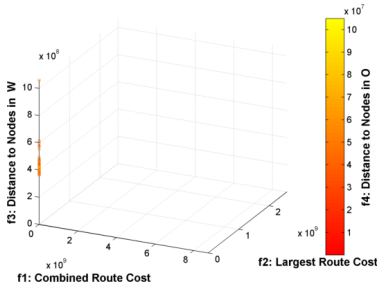
Random Search 0



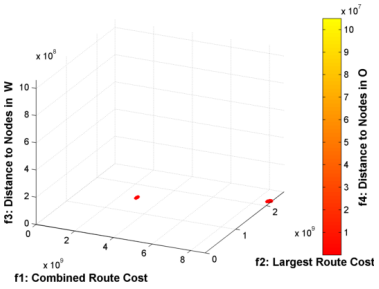
AbYSS 3



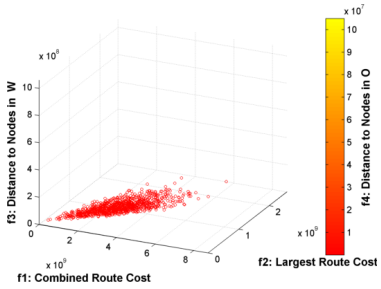
NNRW 1



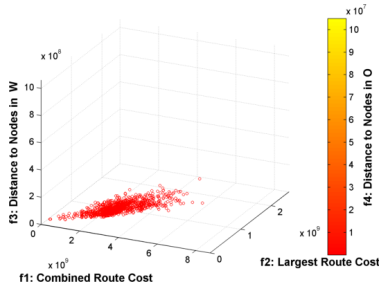
SPEA2 4



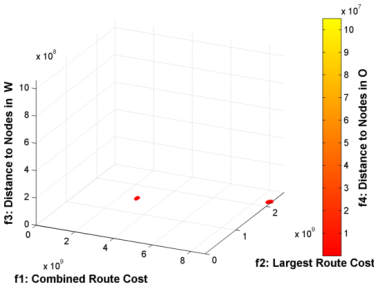
SPEA2 5



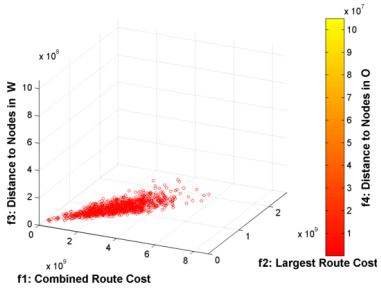
SPEA2 6



NSGA-II 7



NSGA-II 8



NSGA-II 9

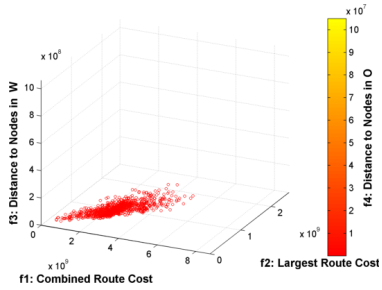
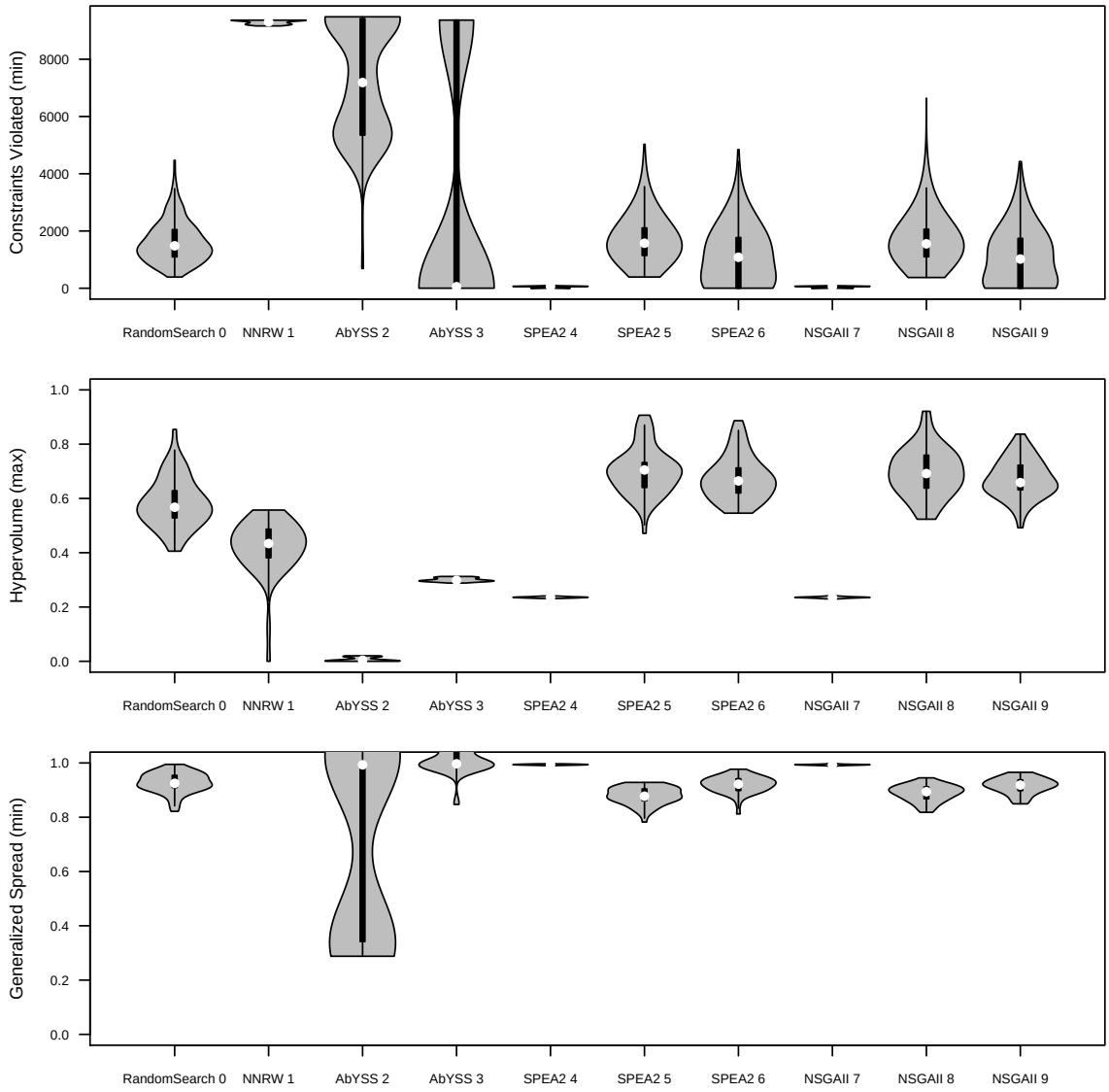
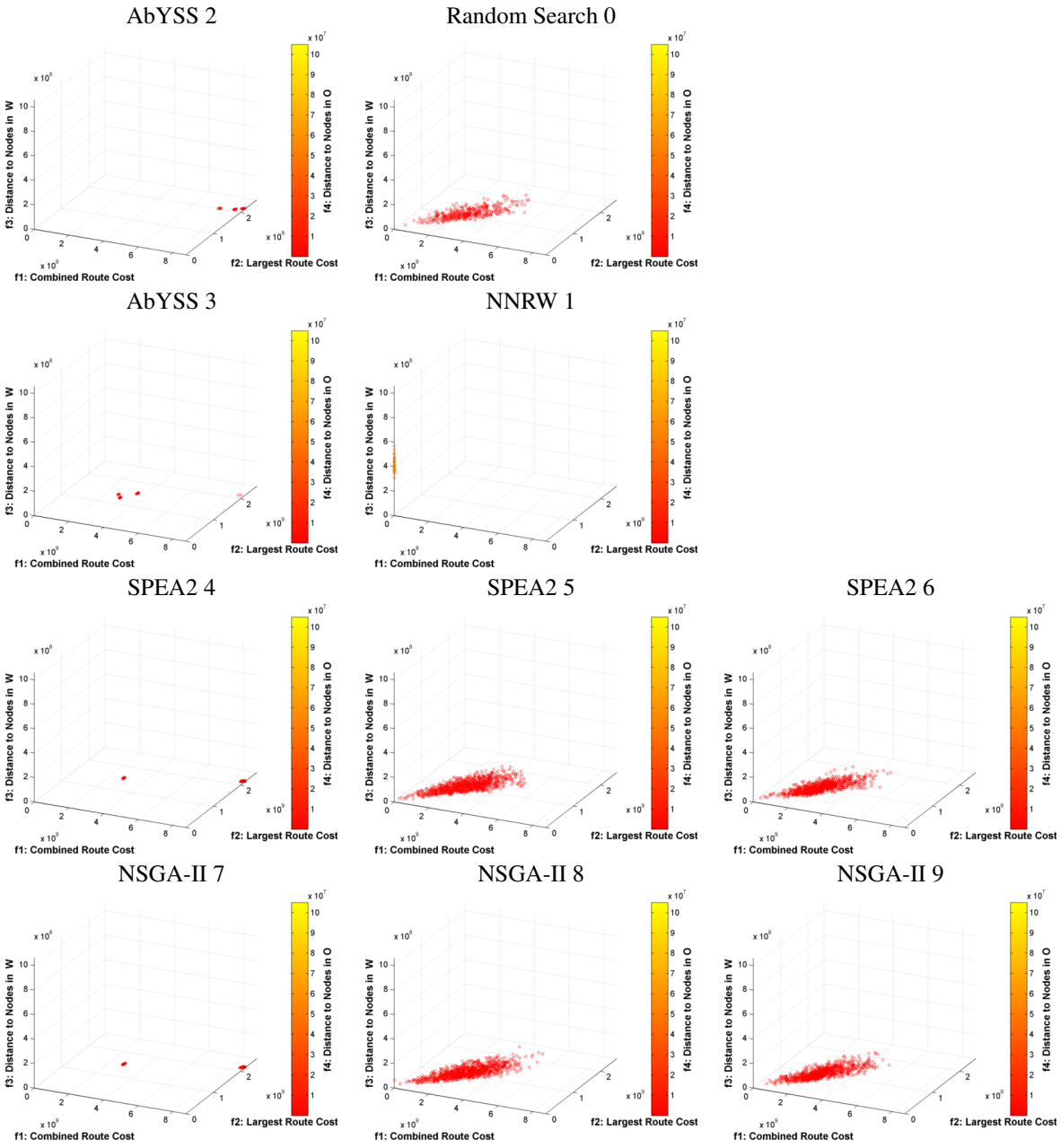


Figure 5.11: USA 13509 - Moderate EDOD - Violin Plots



Short Name	Solution Type	Crossover	Insertion	Deletion	Repair
RS 0	Variable	-	-	-	None
NNRW 1	Variable	-	-	-	None
AbYSS 2	Fixed	-	-	-	None
AbYSS 3	Fixed	-	-	-	Full
SPEA2 4	Fixed	SBX	-	-	Full
SPEA2 5	Variable	RBX	Unif. Addition	Unif. Deletion	None
SPEA2 6	Variable	RBX	Unif. Addition	Unif. Deletion	Full
NSGAII 7	Fixed	SBX	-	-	Full
NSGAII 8	Variable	RBX	Unif. Addition	Unif. Deletion	None
NSGAII 9	Variable	RBX	Unif. Addition	Unif. Deletion	Full

Figure 5.12: USA 13509 - Moderate EDOD - Pareto Fronts



5.3 Algorithm-Level Analysis

The baseline algorithm Random Search (RS) 0 has consistently poor number of constraints violated values across all problems. It is also among the best in hypervolume

and spread. This is likely due to the sheer number of solutions it is able to produce in the time limit and the unbiased fashion it constructs them. Other than Nearest Neighbor with Random Weights (NNRW) the other methods only keep 20 solutions at a time (some keep a larger number in their archive) limiting their generalized spread values, and potentially trapping them in local optima which could affect their hypervolume. Although RS has high hypervolume values it also created a larger number of infeasible solutions.

NNRW 1 is among the worst with respect to the number of constraints violated. It has a more mixed performance for hypervolume. While the bulk of the solutions it produces are in the 0.6 to 0.8 range, it seems to suffer from a long tail which drags its solution sets down. Considering that competing methods show more consistent performance and its poor constraints violated values, it is likely a poor recommendation. Note NNRW 1 is the only algorithm which appears different in the epsilon metric provided in Appendix Appendix A versus hypervolume. Its aggregated solution set plot shows this is likely due to how its solutions are distributed in a narrow portion of the objective space. While they show excellent f_1 and f_2 values, they perform poorly with respect to f_3 and f_4 and have few trade off solutions. In addition, its constraint violation values keep it from being a recommendation.

Archive-based Scatter Search (AbYSS) 2 has consistently bad constraints violated values. Worse still, it has the worst hypervolume values across all contenders for all problems. Its only positive aspect are generalized spread values for the USA problems. However, this does not carry over to the Uruguay or Canada problems. However this suggests its solutions are clustered, which is not necessarily good given their utility values. Furthermore, the location of "good" solutions in objective space may drift apart as the problem size increases meaning proximity to other solutions is not necessarily a good attribute in all cases. Taken as a whole, AbYSS 2 appears to be poor choice in most respects.

AbYSS 3 improves upon AbYSS 2's constraints violated performance, but not by much with respect to the other methods. Its hypervolume values seem to be consistently in the 0.35 to 0.4 range which is comparable to Strength Pareto Evolutionary Algorithm (SPEA) 4 and Nondominated Sorted Genetic Algorithm II (NSGA-II) 7, but not the best by far. Its long tail in generalized spread for the largest problems is still present, but overall its spread is average to poor among the smaller problems. This makes sense as the only difference between AbYSS 2 and 3 are AbYSS 3's inclusion of the full repair operator. This may explain its improved constraint violation performance but seems to have considerably detracted from its hypervolume and spread numbers. Due to the way AbYSS uses value frequencies along with their location in the solution, the repair operators constant shuffling of variables may conflict and create degraded performance but increased feasibility.

Strength Pareto Evolutionary Algorithm 2 (SPEA2) 4 and NSGA-II 7 are similar in performance. They both have low constraint violations. For the two larger problems (USA and Canada) they are actually the two best with respect to the constraints violated metric. While they both are among the worst in generalized spread for all problems, they show middling performance in hypervolume for every problem (0.35 to 0.4 much like AbYSS 3). Depending on the DM, these may be a top choice due to their solutions' feasibility, albeit with a poorer utility.

SPEA2 5 and NSGA-II 8 are also similar in performance. Both use a variable length solution encoding unlike SPEA2 4 and NSGA-II 7. They also both use route based crossover (RBX), Uniform addition, Uniform deletion and no repair operator. This seems to dramatically increase the variance among solutions, including the full range of possibilities when it comes to constraint violations over all problems. They are also notable in that they are among the best with respect to hypervolume for all problems, roughly matching that of RS. They also show the best or average values for generalized spread across all problems; again showing strong similarities to RS. This may be due to the shared variable-

length solution initializer and lack of any repair operator, which possibly explains the wide variation in number of constraints violated.

SPEA2 6 and NSGA-II 9 are also similar in performance. Both use variable length solutions and the same crossover, insertion and deletion operators used in SPEA2 5 and NSGA-II 8. The only difference is that SPEA2 6 and NSGA-II 9 use the full repair operator. With respect to constraints violated, they are among the best across all problems, bested only by SPEA2 4 and NSGA-II 7 which use fixed length solutions, SBX and no insertion or deletion operators. They are also among the best in hypervolume, this time besting SPEA2 4 and NSGA-II 7. Note RS, SPEA2 5, and NSGA-II 8 are arguably better than NSGA-II 6 and SPEA2 9 in hypervolume, with the difference eroding as the problem size increases. In generalized spread, both show average to good performance with SPEA2 6 perhaps having a small lead over NSGA-II 9. Again, this may be explained by the repair operator which seems to limit the wide variance found in SPEA2 5 and NSGA-II 8's constraint violation performance. While the repair operator also seems to have lowered their hypervolume and generalized spread performance, the difference is relatively small.

Depending on the DM SPEA2 6 and NSGA-II 9 would likely be among the top candidate methods due to their all-around excellent performance. The only strong competition comes from SPEA2 4 and NSGA-II 7 which are their fixed-length counterparts which perform much better in constraints violated but are bested with respect to hypervolume and possibly generalized spread.

5.4 Operator-Level Analysis

The metric results suggest that the flexibility offered by variable length solutions, coupled with route-based crossover, insertion and deletion operators provides good spread and hypervolume given the time limit. Alternatively, the fixed length solutions versions of the same algorithms seem to result in poorer solutions with respect to utility, but with fewer violated constraints. Since all of the best methods utilize the full repair method,

this difference in constraint violation performance is likely due to the higher volatility in variable-length methods. The insertion, deletion and route-based crossover can be thought of as providing a large neighborhood for local search. This volatility also seems to allow the methods to escape local optima to obtain higher utility values. For this reason, caution should be taken in making conclusions as to the relative performance of variable versus fixed size solutions. Questions as to the relative importance of the insertion, deletion, crossover and solution type is an open question.

Given that the repair operator is common to what are arguably the best methods (SPEA2 4, NSGA-II 7, SPEA2 6 and NSGA-II 9), we feel the benefits of the operator are clear. This does not allow us determine which parts of the monolithic repair operator are responsible for the performance boost however. It is possible that one or more aspects may hinder it or a different mix of flags would result in better performance.

The aggregate solution set plots suggest that the feasible solution space is continuous and relatively small. Note that the problem generation method used in this work results in "repulsor" nodes from the set O being close to "attractor" nodes in W and T. This results in a highly constrained objective space and possibly explains the small amount of variation in solutions with respect to f_3 and f_4.

These plots also show the repair operator to consistently limit solutions to the assumed feasible region across all algorithms. The design issue may be that the algorithm must be able to escape this local minima within one iteration such that the repair operator does not prematurely cutoff promising solutions.

5.5 Summary

While a single definitive algorithm cannot be identified given the MOP nature of the results, we have identified four algorithms as roughly being the best candidates (two versions of both SPEA2 and NSGA-II). SPEA2 4 and NSGA-II 7 provide excellent performance in number of constraints violated, while their variable-length counterparts

SPEA2 6 and NSGA-II 9 provide worse performance in constraints violated but offer better hypervolume.

From this, we have concluded that the repair operator provides a significant boost to the performance of solvers with respect to number of constraints violated, though the degree which it helps depends on the underlying algorithm assumptions. For example, we believe AbYSS to be a poor fit for our repair operator given its constant shuffling of the solution variables, which AbYSS assumes remain in place.

While the variable-length solution methods outperformed their fixed-length versions in hypervolume and generalized spread, we cannot draw conclusions as to the underlying reason. We believe it to be most likely due to the increased volatility and local search distance the variable length solvers have which provides more opportunities to escape local optima. This is especially important given the repair operator which appears to cutoff promising solutions too soon to escape from local minima. In other words, the repair operator stops all local search after one iteration making that single iteration incredibly important. This hypothesis explains why the less-volatile fixed-length versions are not the ones better in hypervolume as we would expect due to faster operations.

The reasons why fixed-length is better with respect to number of constraints is less clear, though we believe it to be function of the smaller neighborhood it has to search for solutions.

In the end, the best solution method must be chosen by a human DM due to the subjective nature of the analysis. In other words a DM must defined their preferences with respect to the metrics. Even then, given that the data is not Gaussian, choosing a single best is difficult. The importance of variable-length solution encodings and highly volatile operators in cases where hypervolume and generalized spread are important is clear (when paired with the repair operator). Conversely, their fixed-length counterparts are clearly better if number of constraints violated are more important. Finally, the importance of the

repair operate was made clear and the superiority of SPEA2 and NSGA-II with respect to the set tested.

VI. Conclusion

The Dynamic Mulit-Objective Multi-vehicle Covering Tour Problem (DMOMCTP) formalizes the problem of routing multiple cooperating Unmanned Aerial Vehicles (UAVs) to collect intelligence via arbitrary sensors while abiding by several constraints and optimizing several objectives. This is meant to model surveillance targets, hostile forces and the competing goals of the mission planner who ultimately decides which plan to implement.

While the number of works related to the DMOMCTP is large, the DMOMCTP combines several components that have been studied extensively into a problem that is new and that no other work has addressed to our knowledge. This research thus briefly covers works which address one or more of the individual components included in the DMOMCTP and the solvers studied.

This work introduces the DMOMCTP formalization and studies the efficacy of several centralized solvers with respect to a hard time limit. The goal being to find the best set from those tested with respect to the DMOMCTP. This allows future works - whether they be on centralized or decentralized solvers - to focus on promising algorithms.

This work approached the DMOMCTP from an a posteriori perspective requiring the resulting sets of solutions to be measured via several metrics. This study used hypervolume, generalized spread and number of constraints violated while providing several others in Appendix Appendix A. This set covers the major desirable qualities and showed the greatest amount of variance among the solvers studied.

The problem instances are modified from Traveling Salesman Problem (TSP) data sets which were themselves from real-world Geographic Information System (GIS) data sets. These are adapted to DMOMCTP problem instances. Three countries are used for the study, each with two levels of EDOD resulting in six problem instances. Each solver

is given three minutes to solve each problem per independent run. The experiment itself is run in a distributed fashion. Each algorithm is run over each problem instance fifty times with a time limit of three minutes to return the best set of solutions possible..

6.1 Results

Among the set of algorithms tested, AbYSS, NNRW and RS were the worst performers with NSGA-II and SPEA2 being the best. The fixed-length versions of NSGA-II and SPEA2 performed the best with respect to constraints violated, especially as the problem size increased. However the biggest takeaway is the excellent performance from the variable-length variants of NSGA-II and SPEA2 which were among the best in each metric except for constraints violated in the largest problem. Finally, the repair operator was found to make a significant improvement to an algorithms performance in constraints violated while slightly degrading its generalized spread and hypervolume.

6.2 Significance of Study

This work provides a new problem formalization - the DMOMCTP - and recommends a subset of the algorithms studied for further analysis and use in time constrained use cases. This allows future works to focus on promising algorithms while including the best from this work for comparison. This work also provides insight into operator design with respect to the DMOMCTP, namely the apparent importance of repair operator design, the use of variable-length operators and solution encodings. It finds NSGA-II and SPEA2 to be the best algorithms given a three minute time limit, besting versions AbYSS, NNRW and RS.

6.3 Limitations of Study

The native problem domain is inherently complex and difficult to solve which was the impetus for limiting our formalization's scope. Issues related to real-world robotics such as navigation, communication, and physics have been jettisoned and are left for later works. The resulting formalization is still an interesting and difficult problem that we hope

provides a stepping stone to more complete problems. Specifically, we have chosen to focus on central solvers which we hope provides an important baseline for future works.

Furthermore, this work only provides results for fifty independent runs for each algorithm over six problem instances - largely due to the real-world time required for computation. As in other MOPs the nature of this problem and low number of data points, the resulting data is not normal. The problem instances themselves are taken from real-world GIS data sets, but the nodes are assigned to the various sets using a Uniform Random Variable (RV) which is an assumption that may not hold in all problem instances.

6.4 Future Work

Promising areas of inquiry include the decentralized solvers and/or problems which model real-world communications. Also, defining the geographic area over a Euclidean plane, while increasing the degrees of freedom, would provide a more robust and realistic model for the native problem domain. Similarly, formulations that include mobile hostile adversaries and realistic DM preferences would prove interesting and useful.

On an implementation level, studies on the individual operators and their relative effects with respect to individual metrics (e.g. RBX vs SBX with respect to hypervolume) would be useful. Work on paramater tuning, additional time limits, and other algorithms would also be of interest. Implementing the solvers on real UAVs with real-time time limits and hardware constraints would be especially interesting.

6.5 Summary

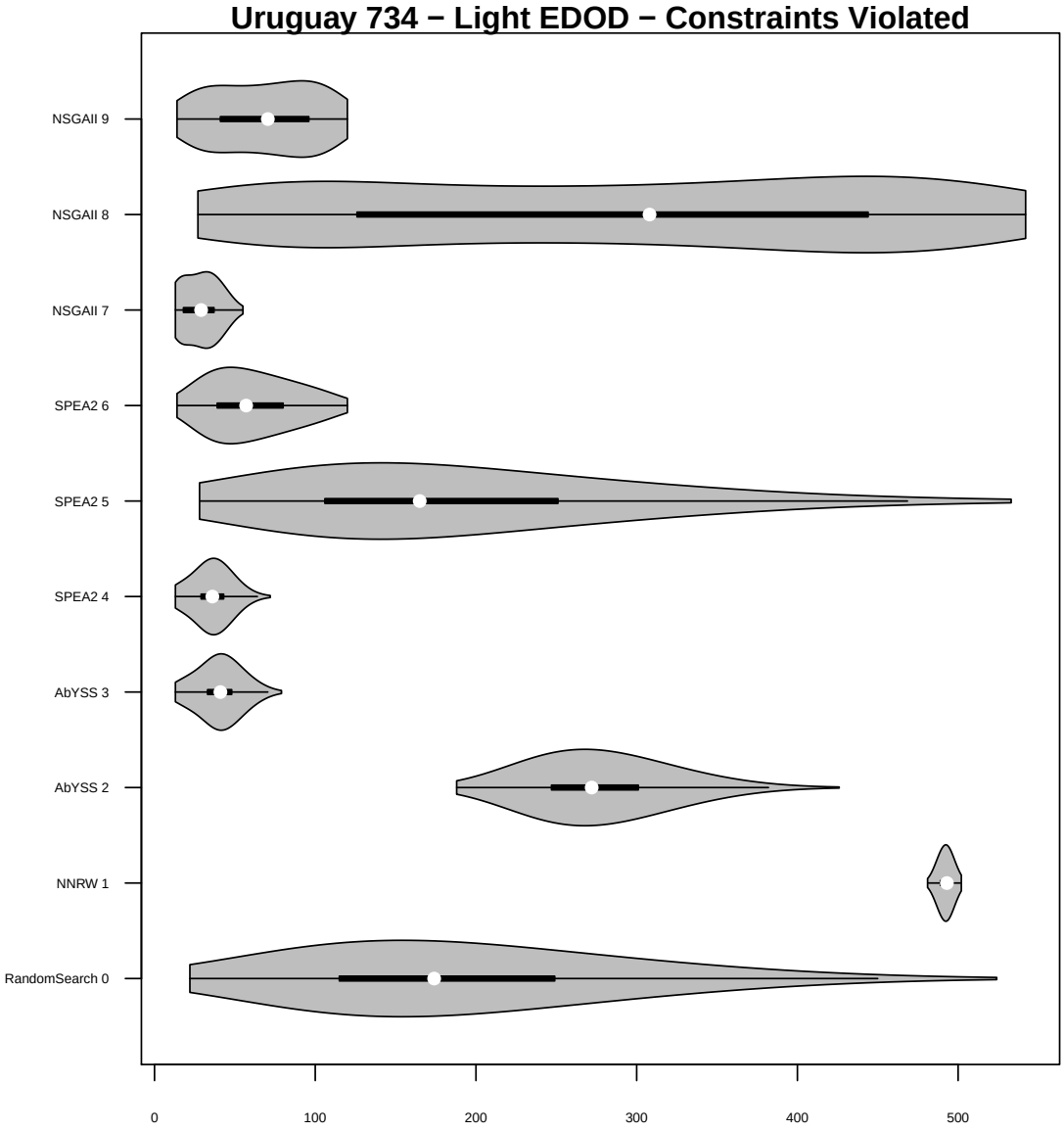
Motivated by a complex, multiobjective, multivehicle routing problem with covering constraints this work introduces a new formalization - the DMOMCTP. As a new problem formalization and given the No Free Lunch Theorem, related works provide limited value in solving this problem. Thus the purpose of this work is to study several centralized solvers against the DMOMCTP and provide a reduced set to recommend for future works and

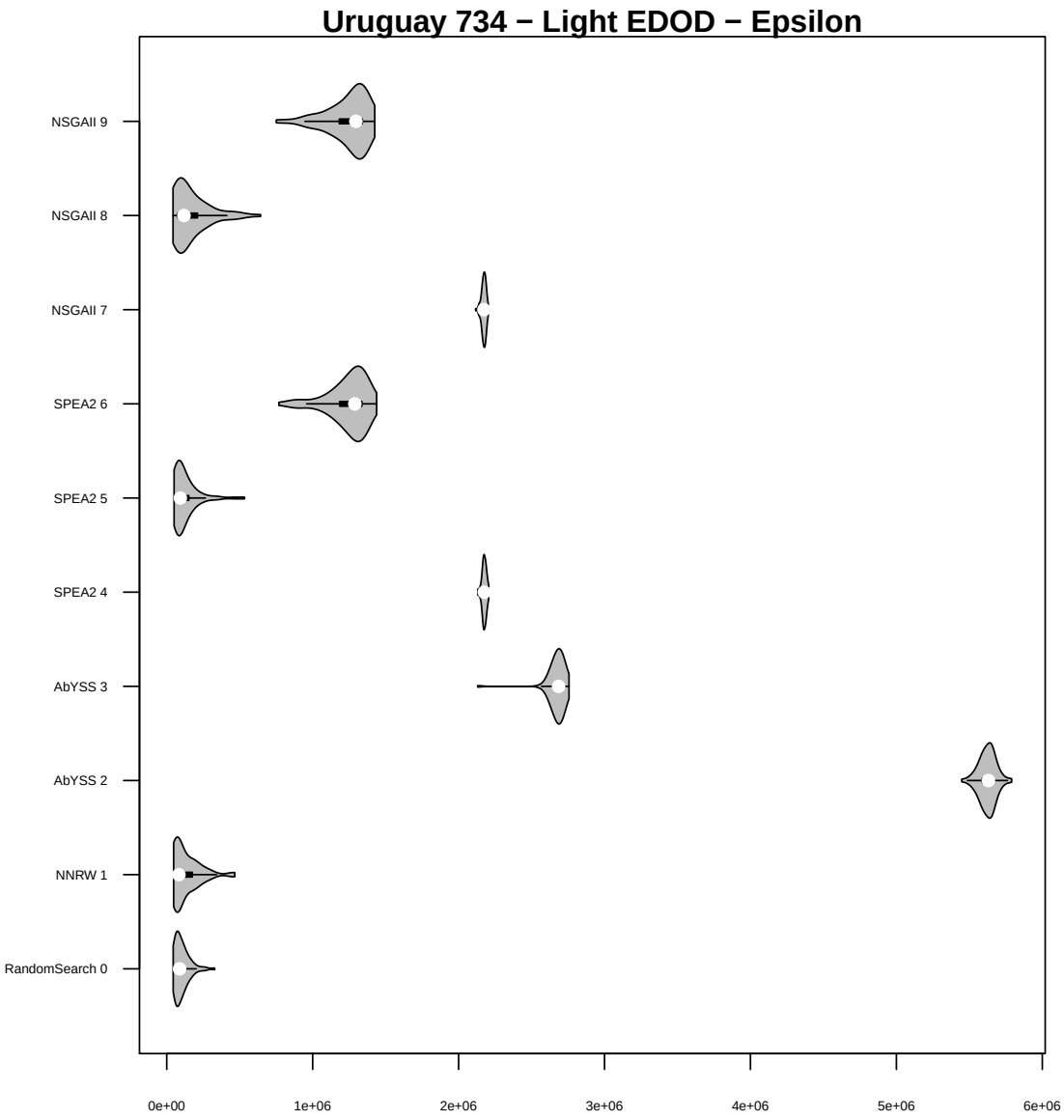
applications. The work finds two to three promising algorithms including variable length versions of NSGA-II, SPEA2, and RS.

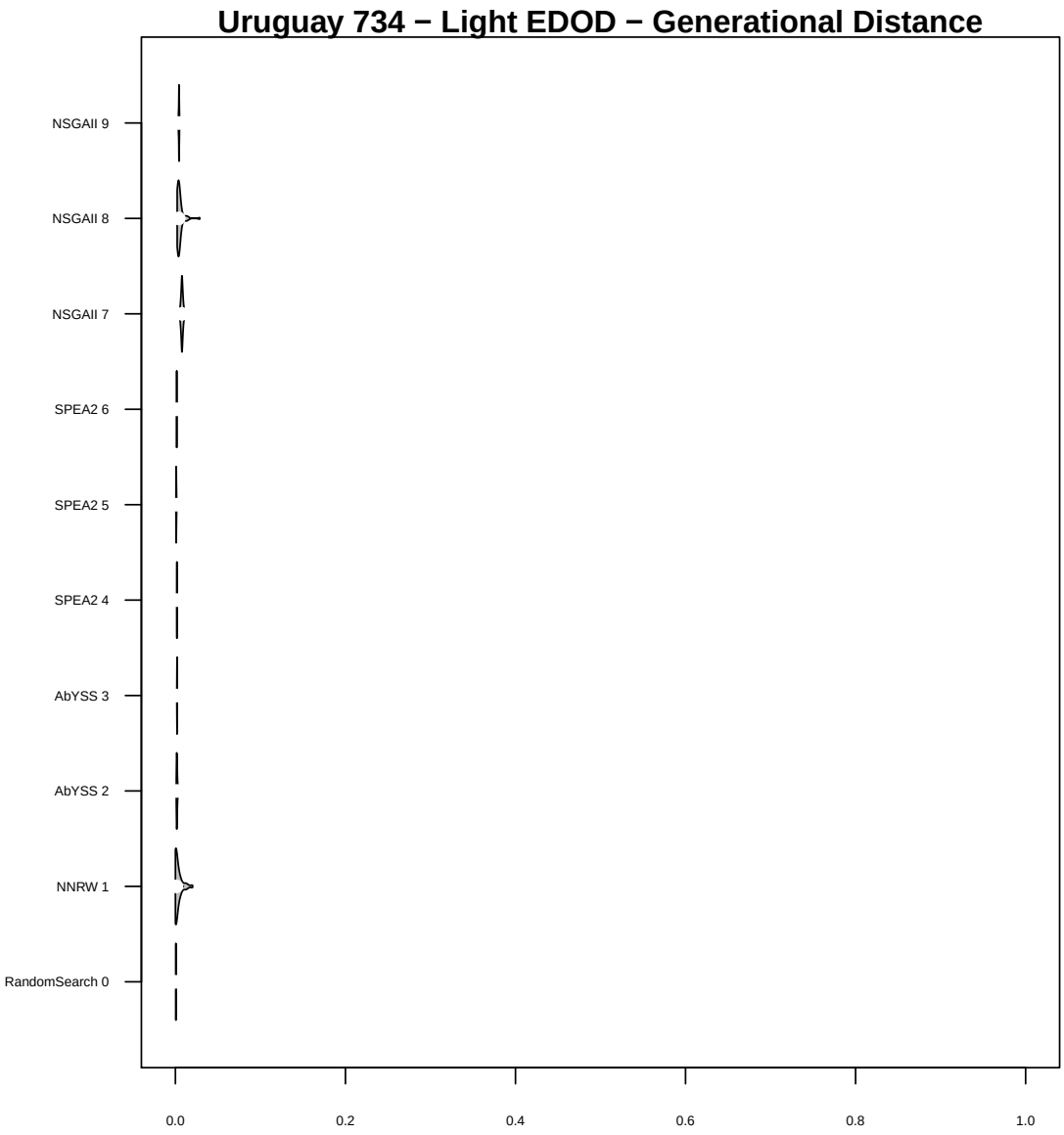
Appendix A: Violin Plots

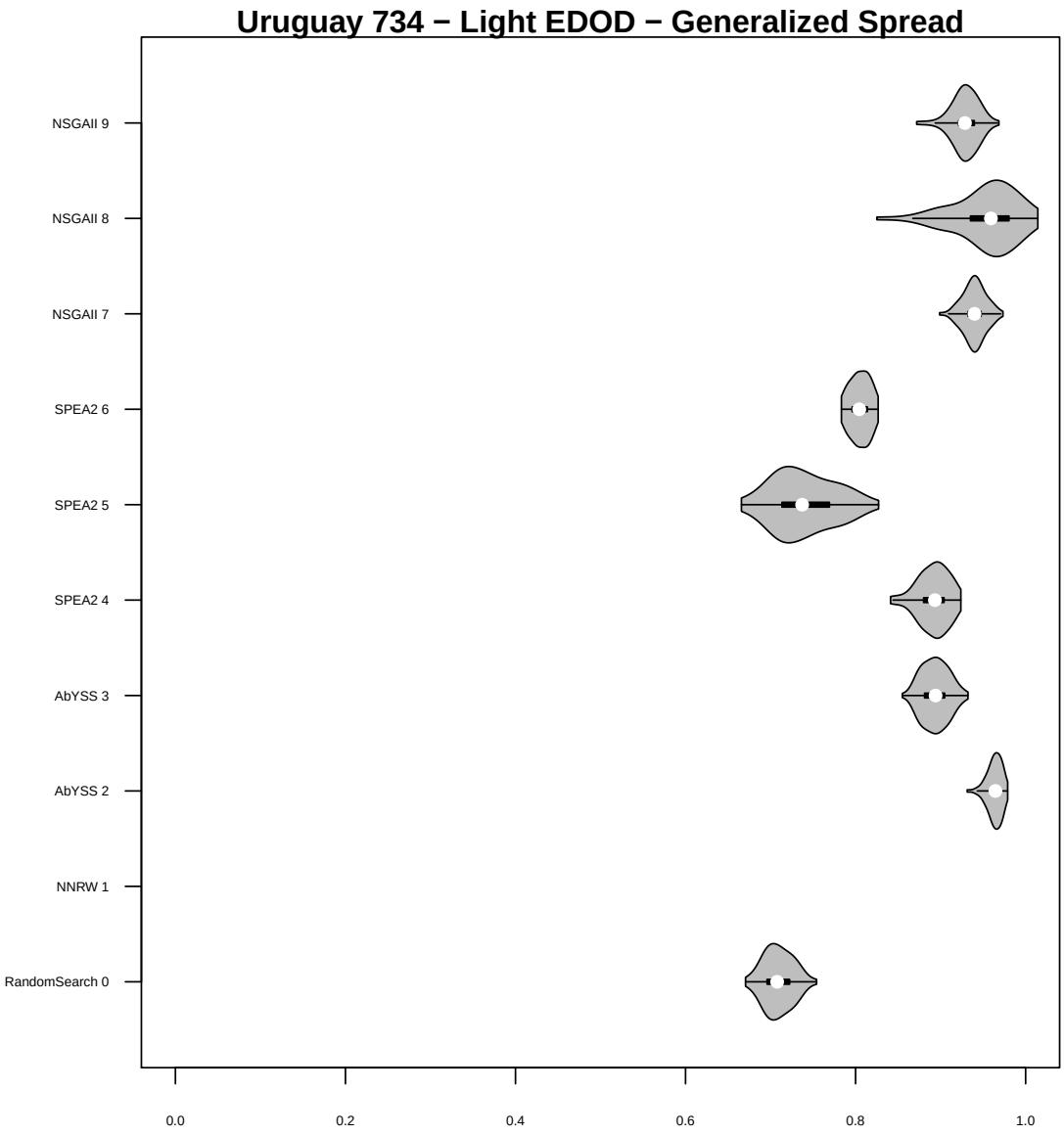
A.1 Uruguay

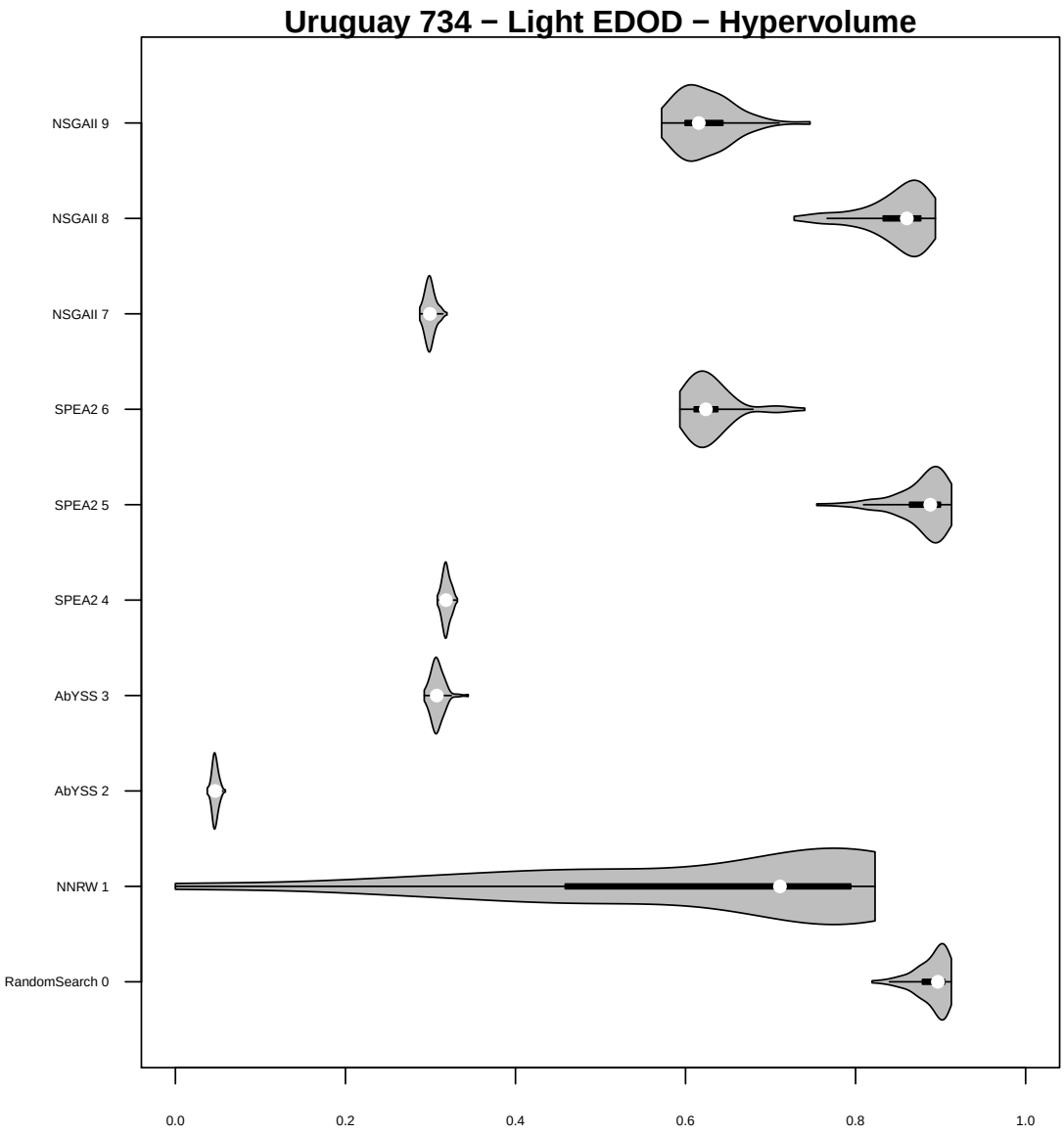
A.1.1 Light EDOD.

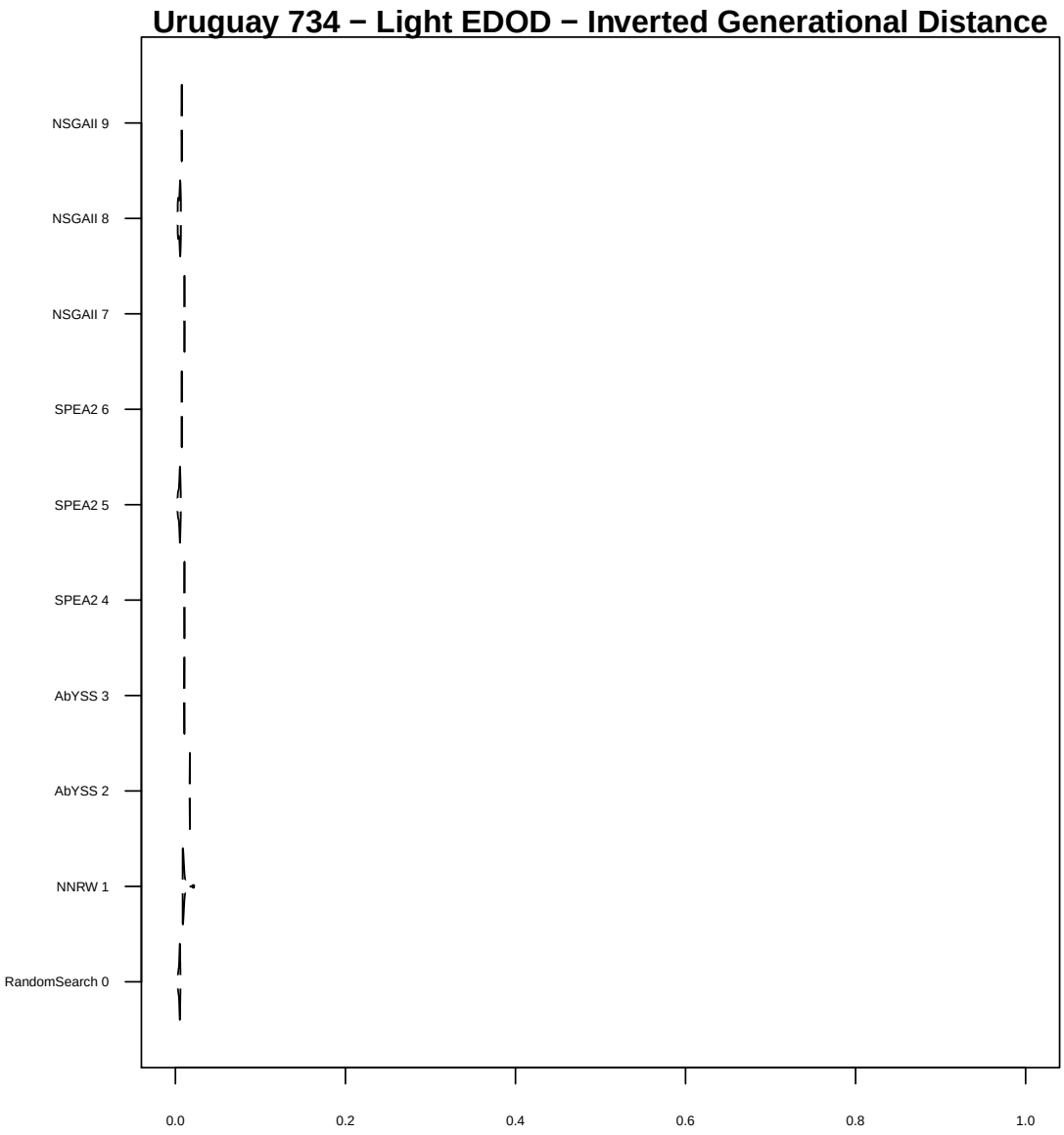



Uruguay 734 − Light EDOD − Epsilon
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06

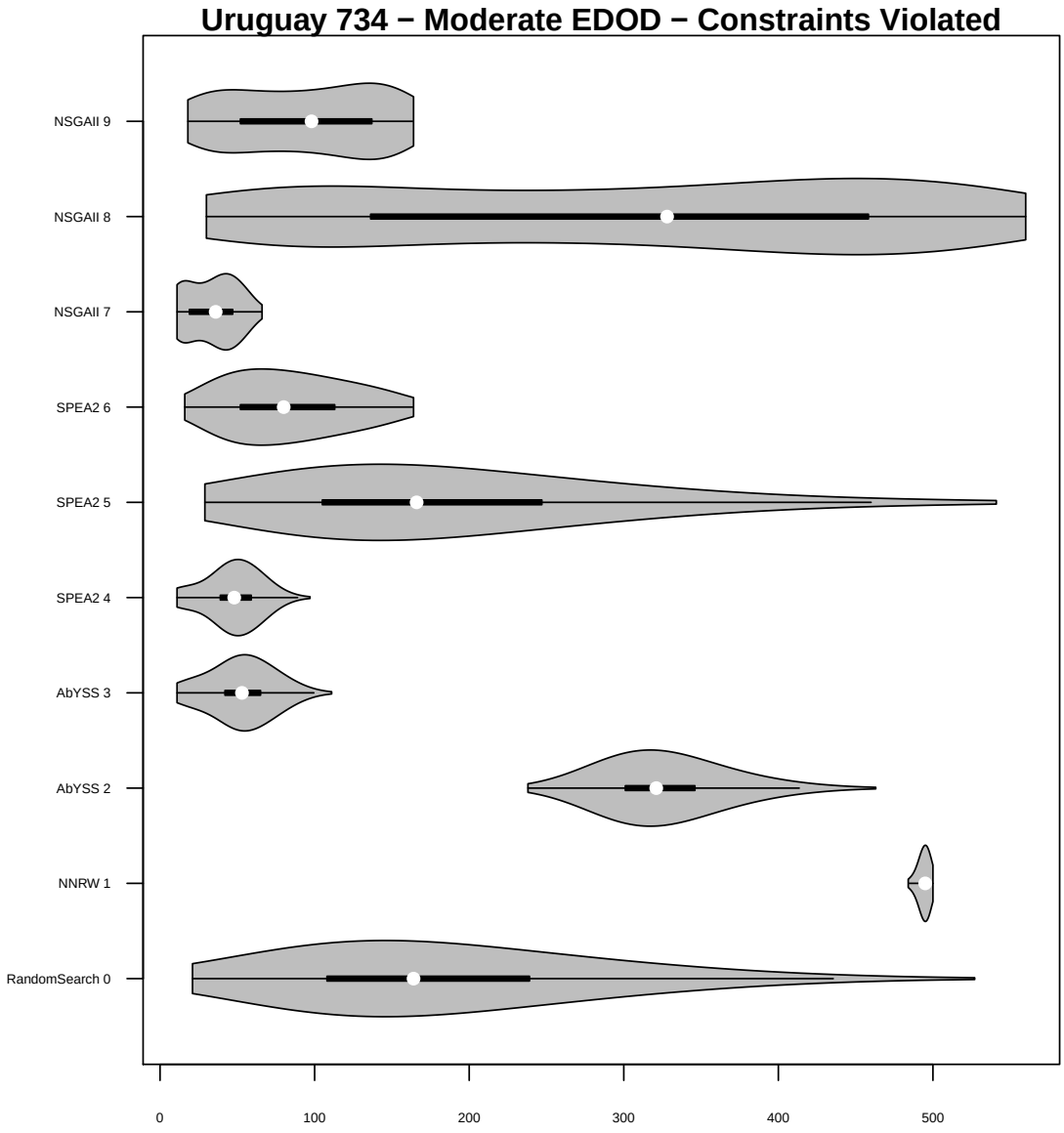

Uruguay 734 − Light EDOD − Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

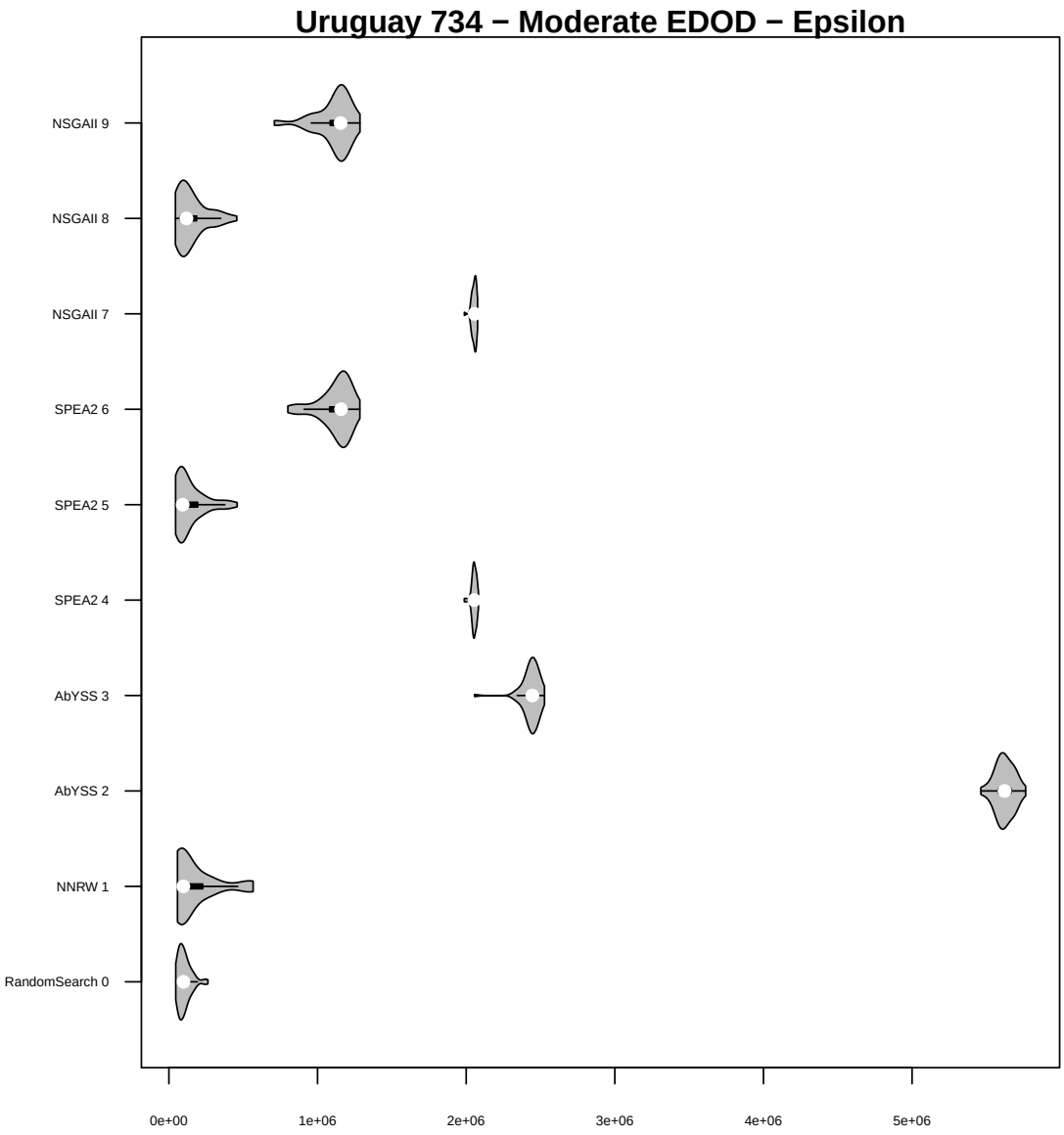

Uruguay 734 − Light EDOD − Generalized Spread
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

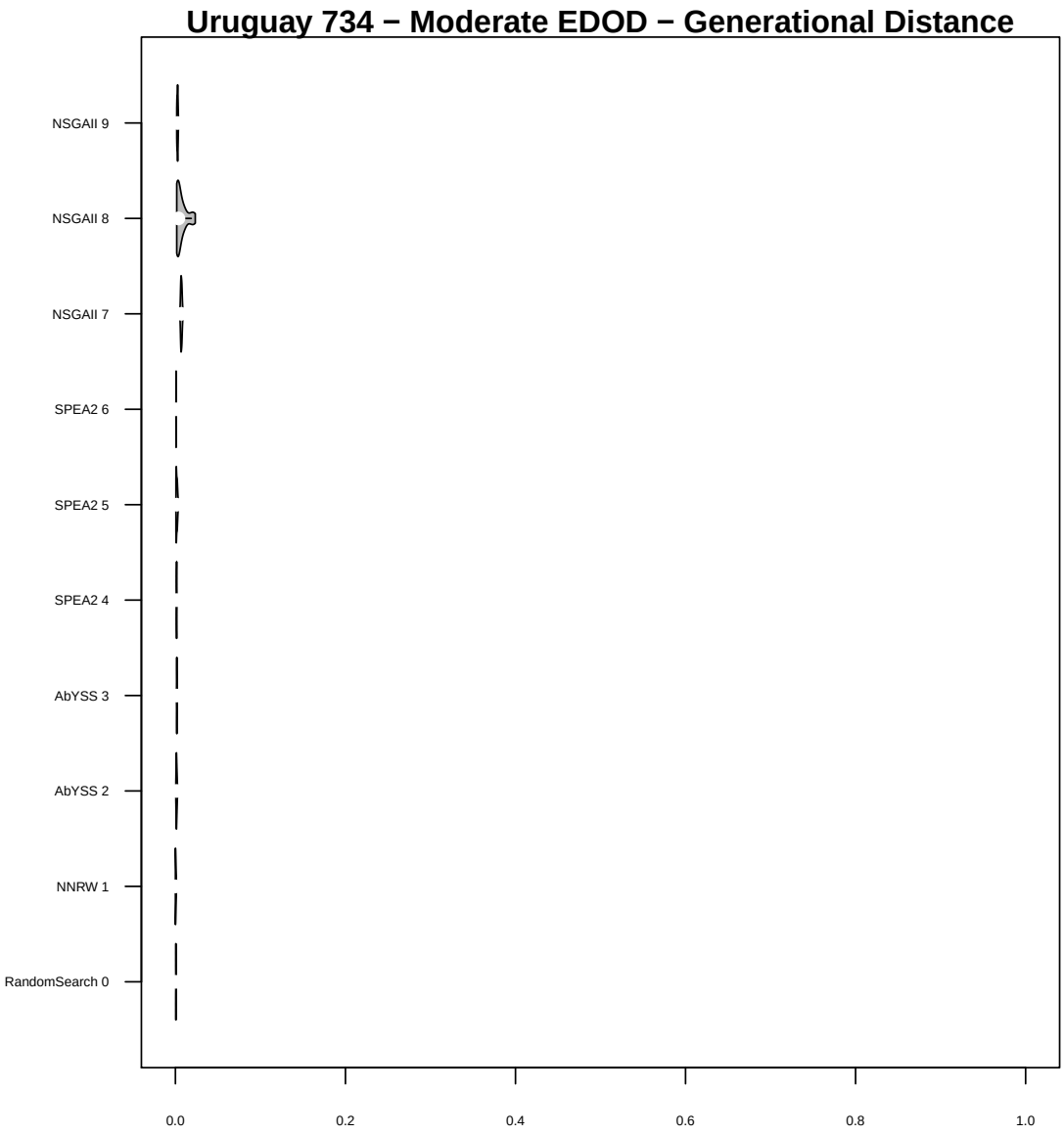

Uruguay 734 − Light EDOD − Hypervolume
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

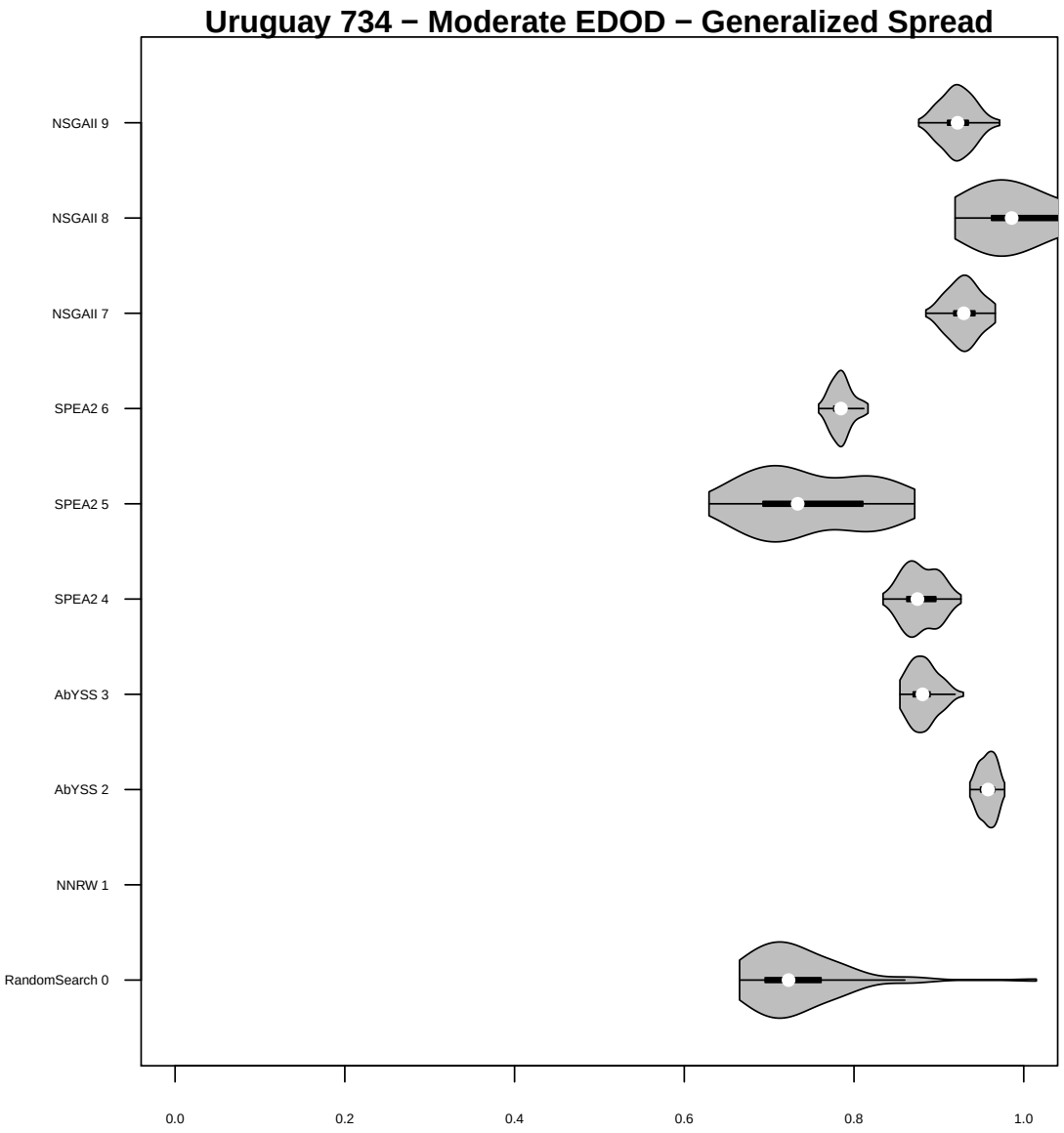

Uruguay 734 − Light EDOD − Inverted Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

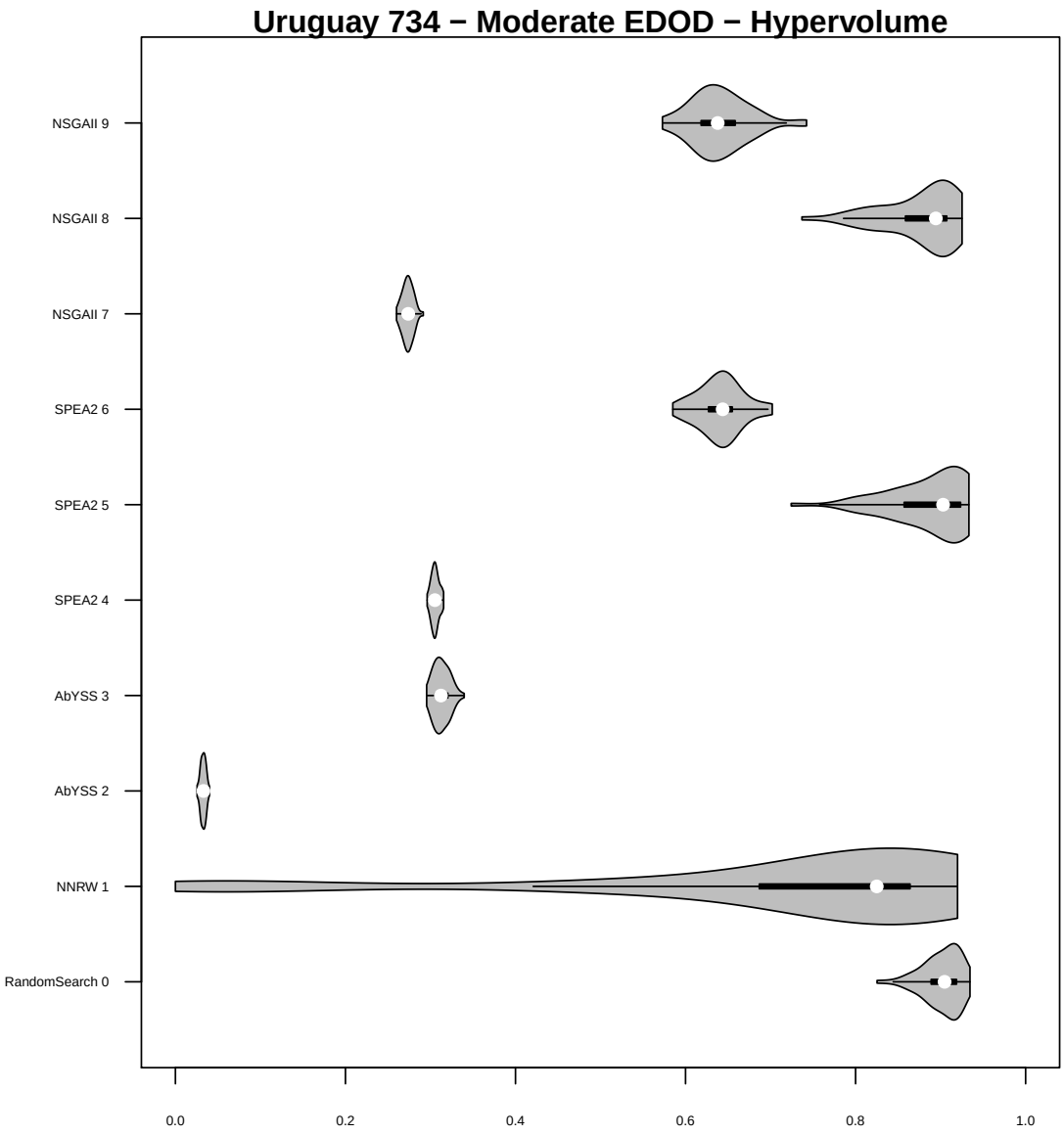
A.1.2 Moderate EDOD.

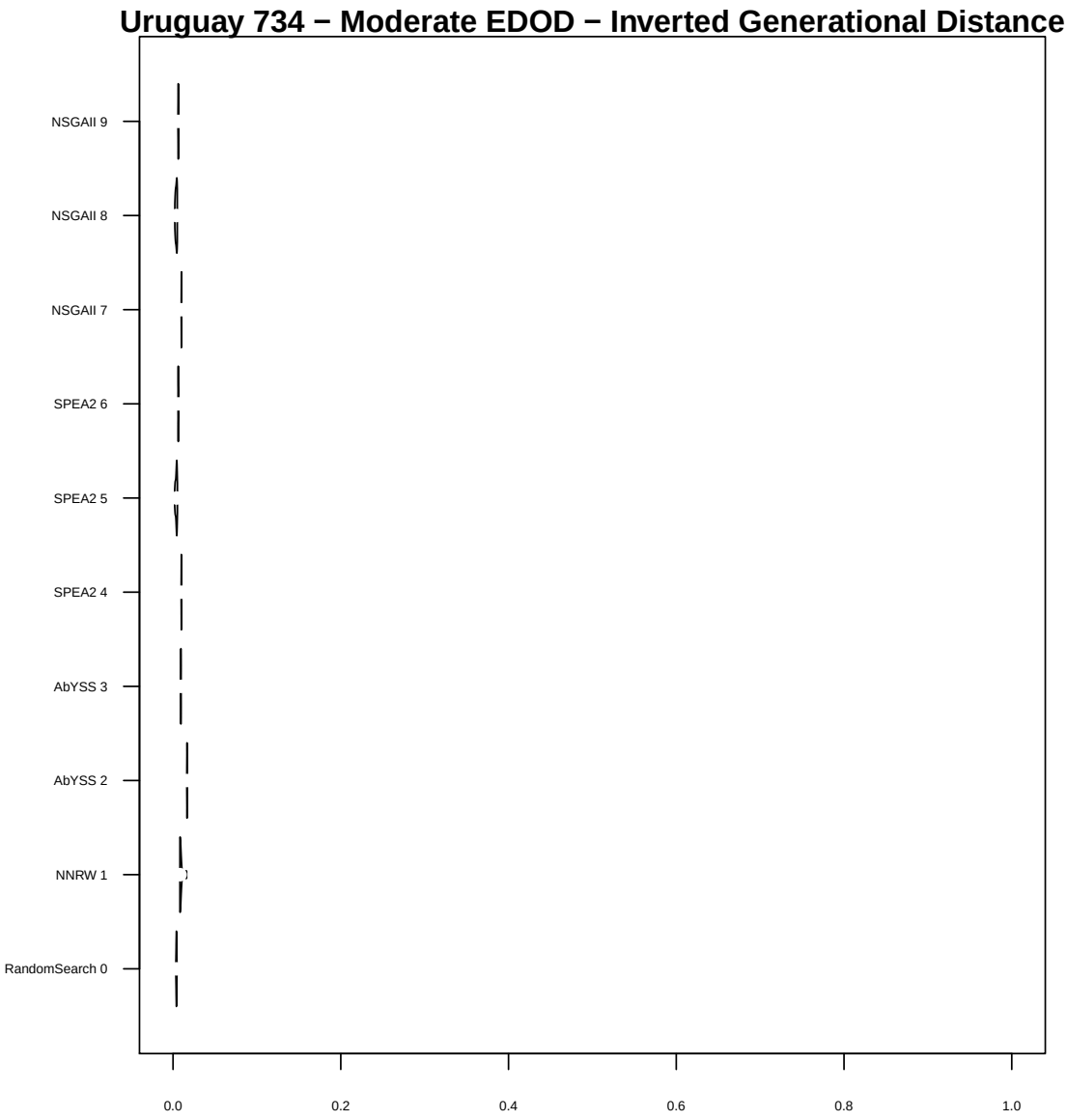



Uruguay 734 − Moderate EDOD − Epsilon
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0e+00
1e+06
2e+06
3e+06
4e+06
5e+06


Uruguay 734 − Moderate EDOD − Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

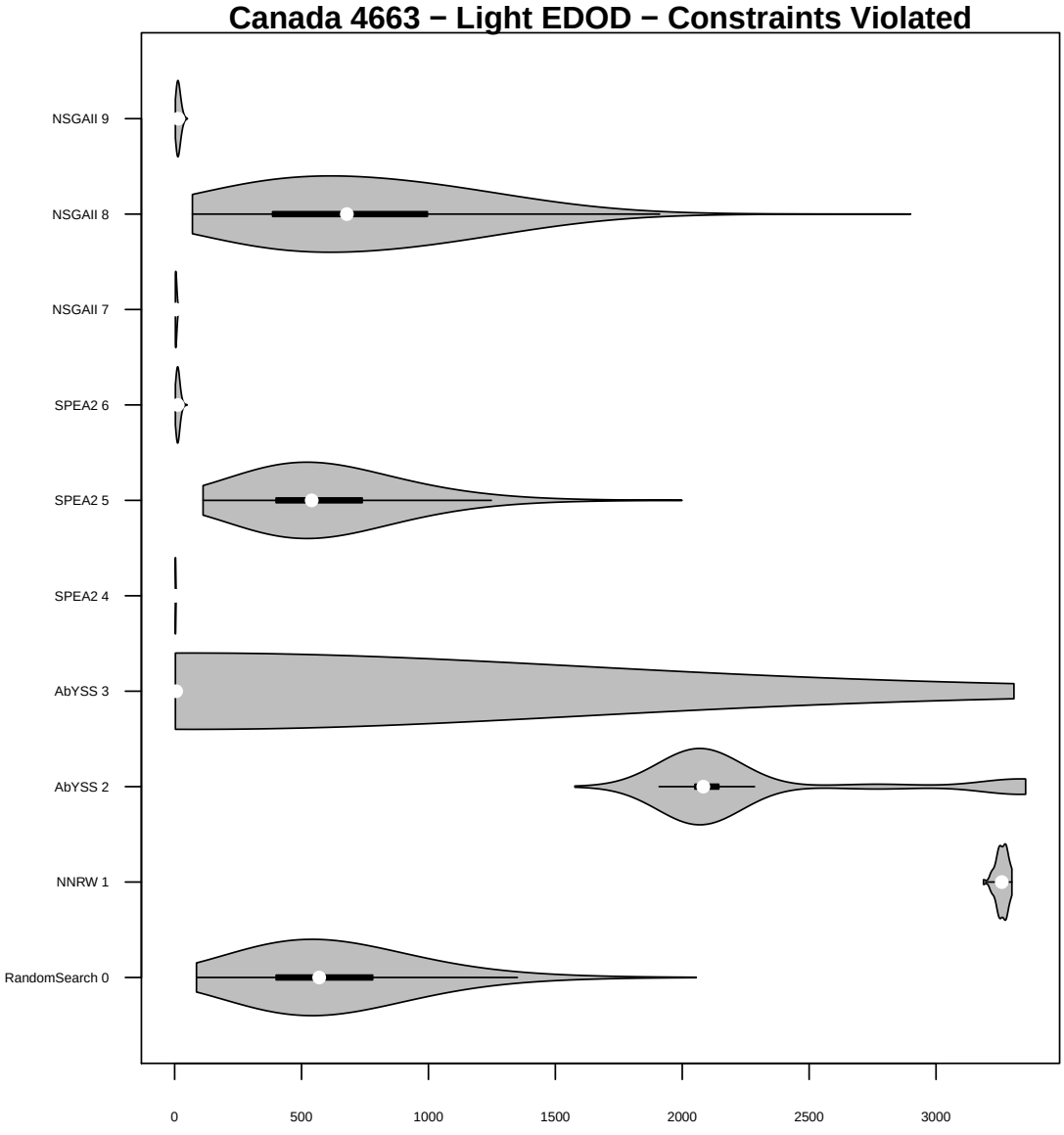

Uruguay 734 − Moderate EDOD − Generalized Spread
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

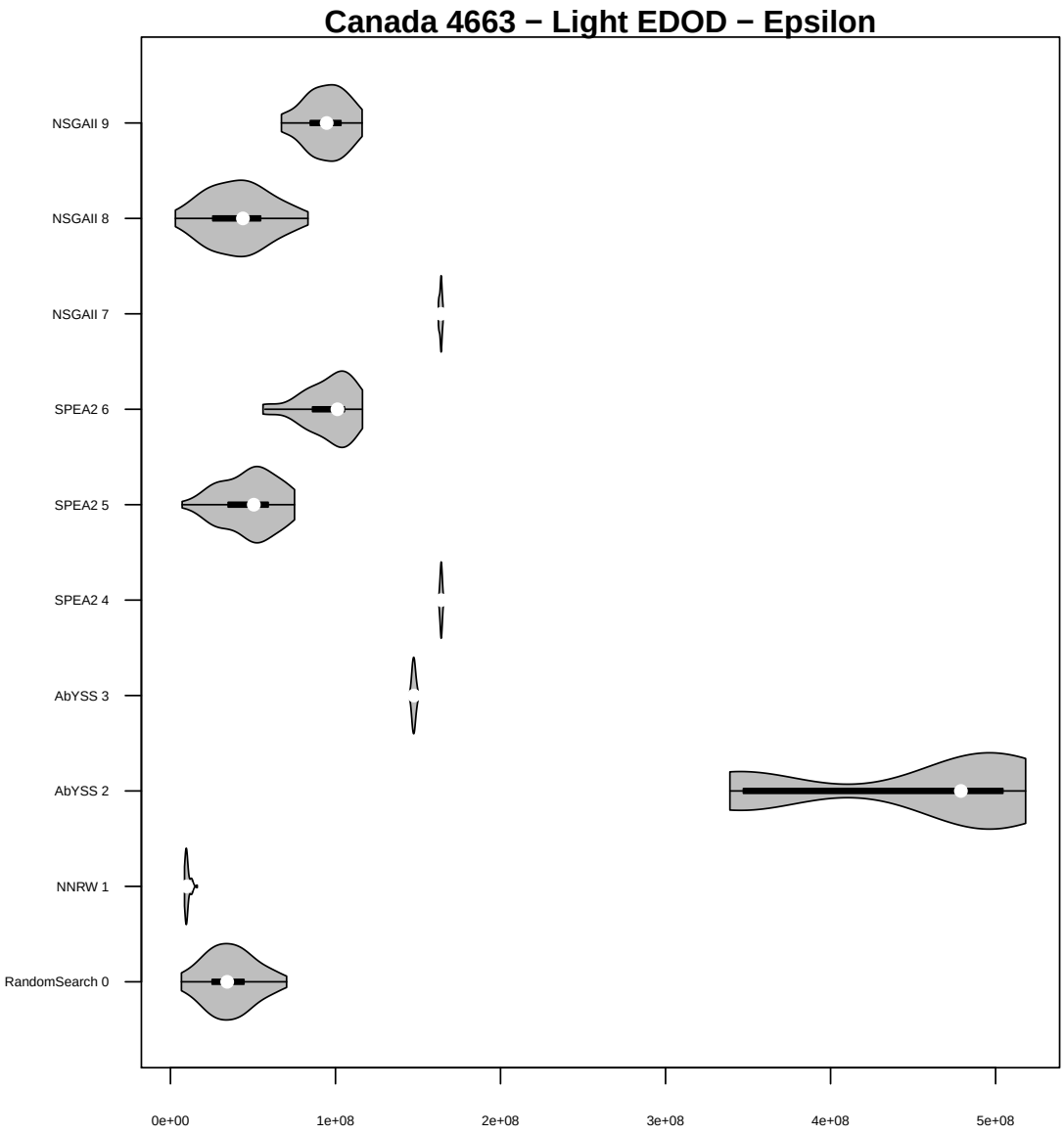

Uruguay 734 − Moderate EDOD − Hypervolume
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

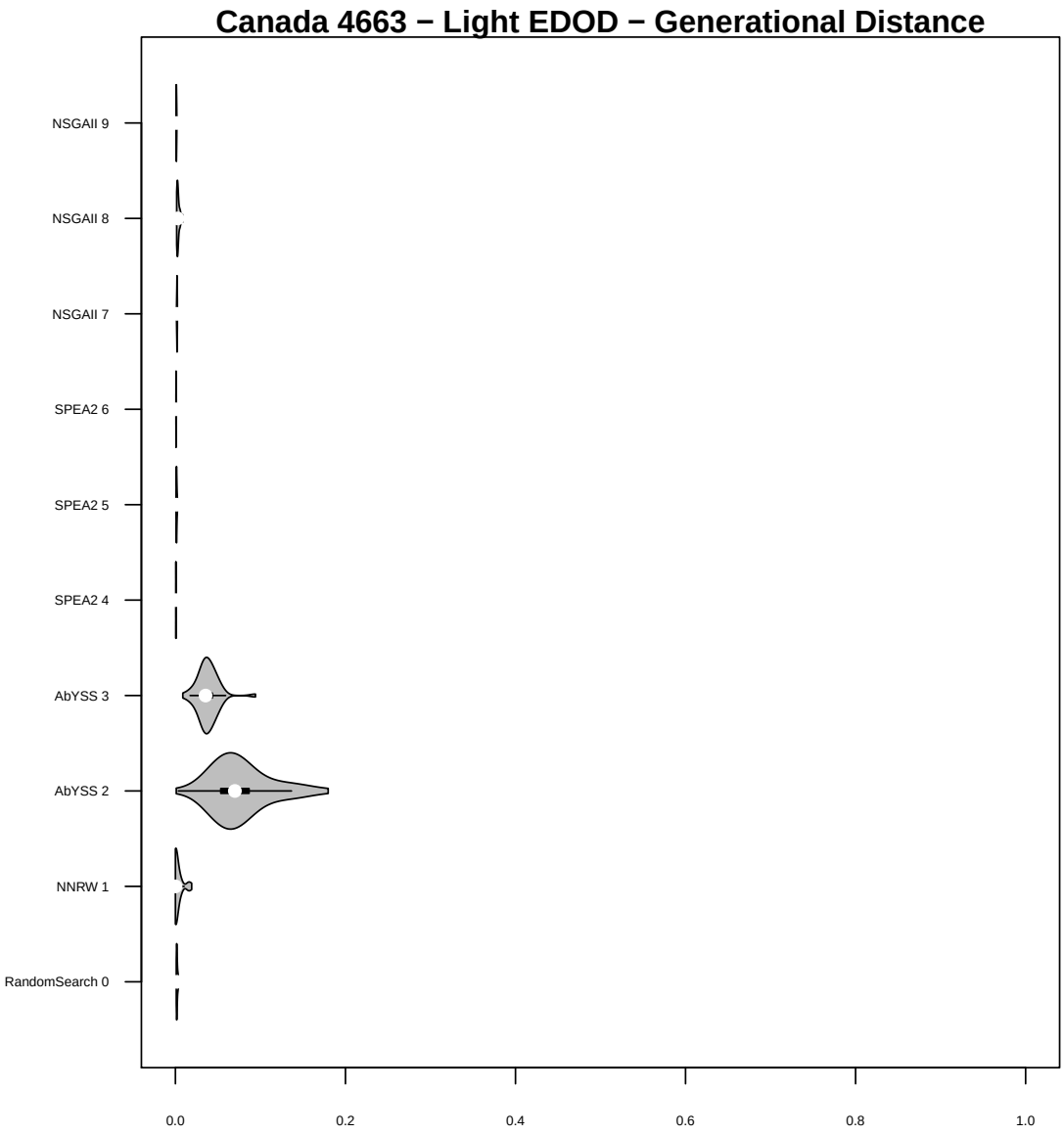

Uruguay 734 − Moderate EDOD − Inverted Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

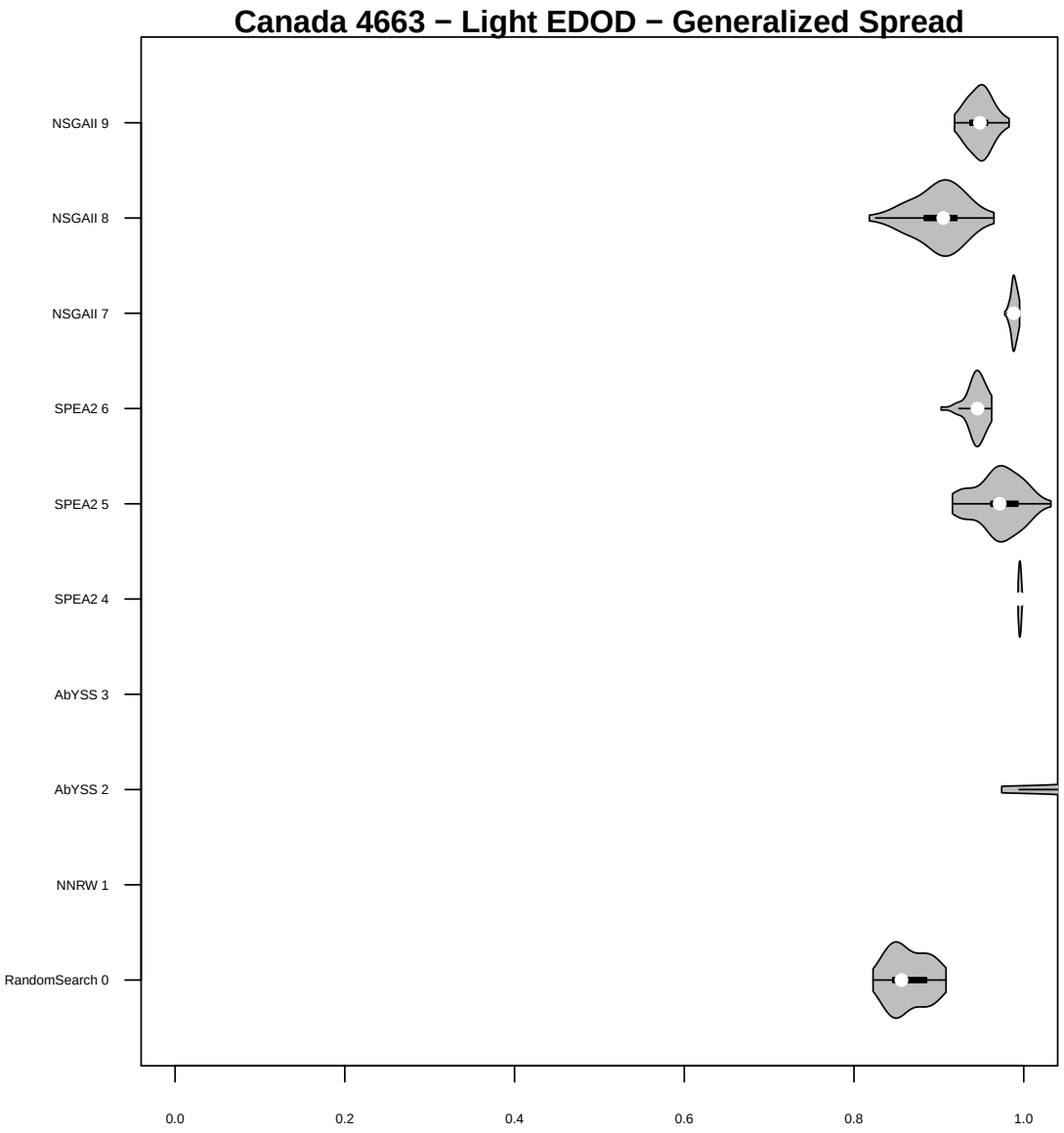
A.2 Canada

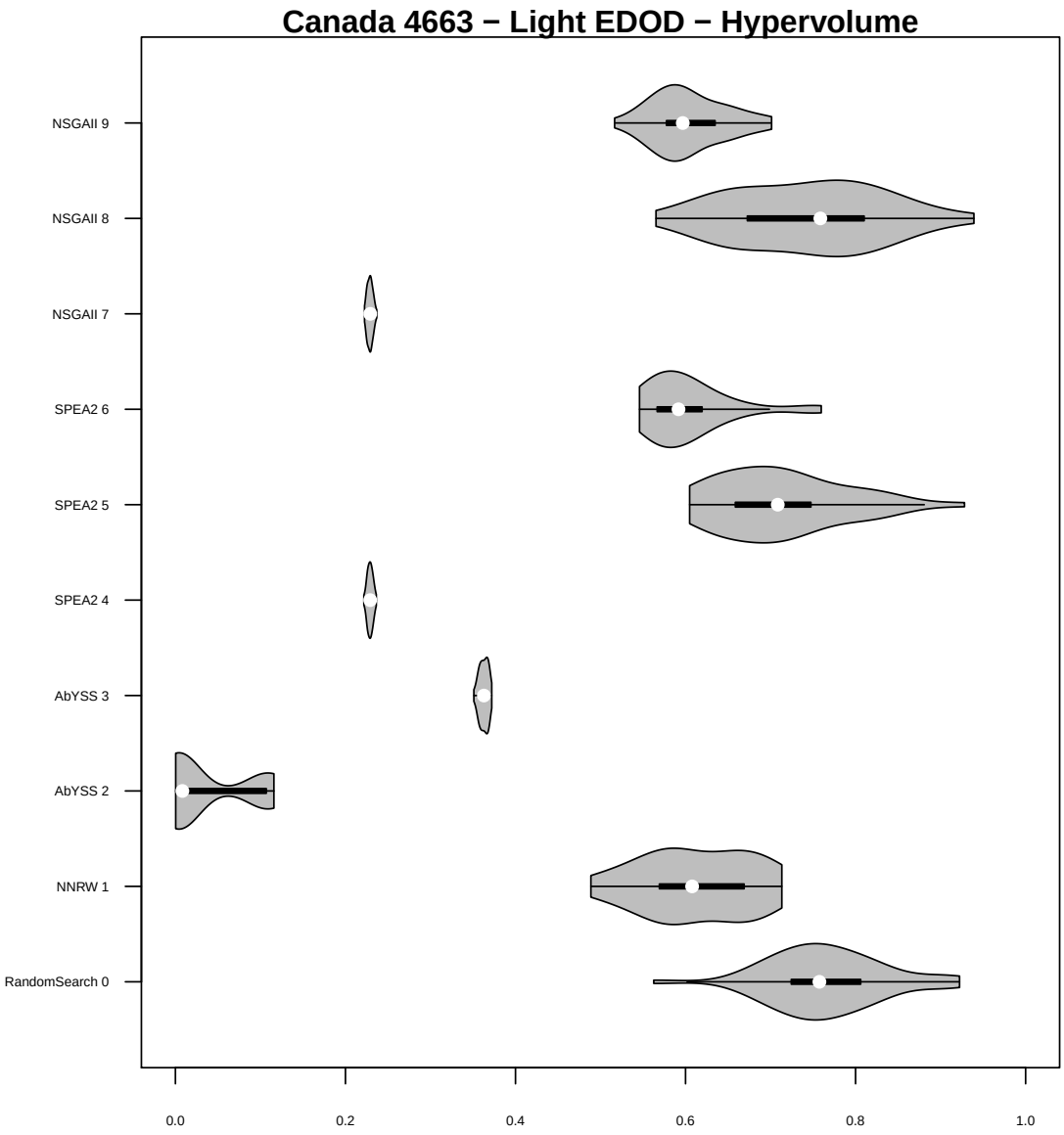
A.2.1 Light EDOD.

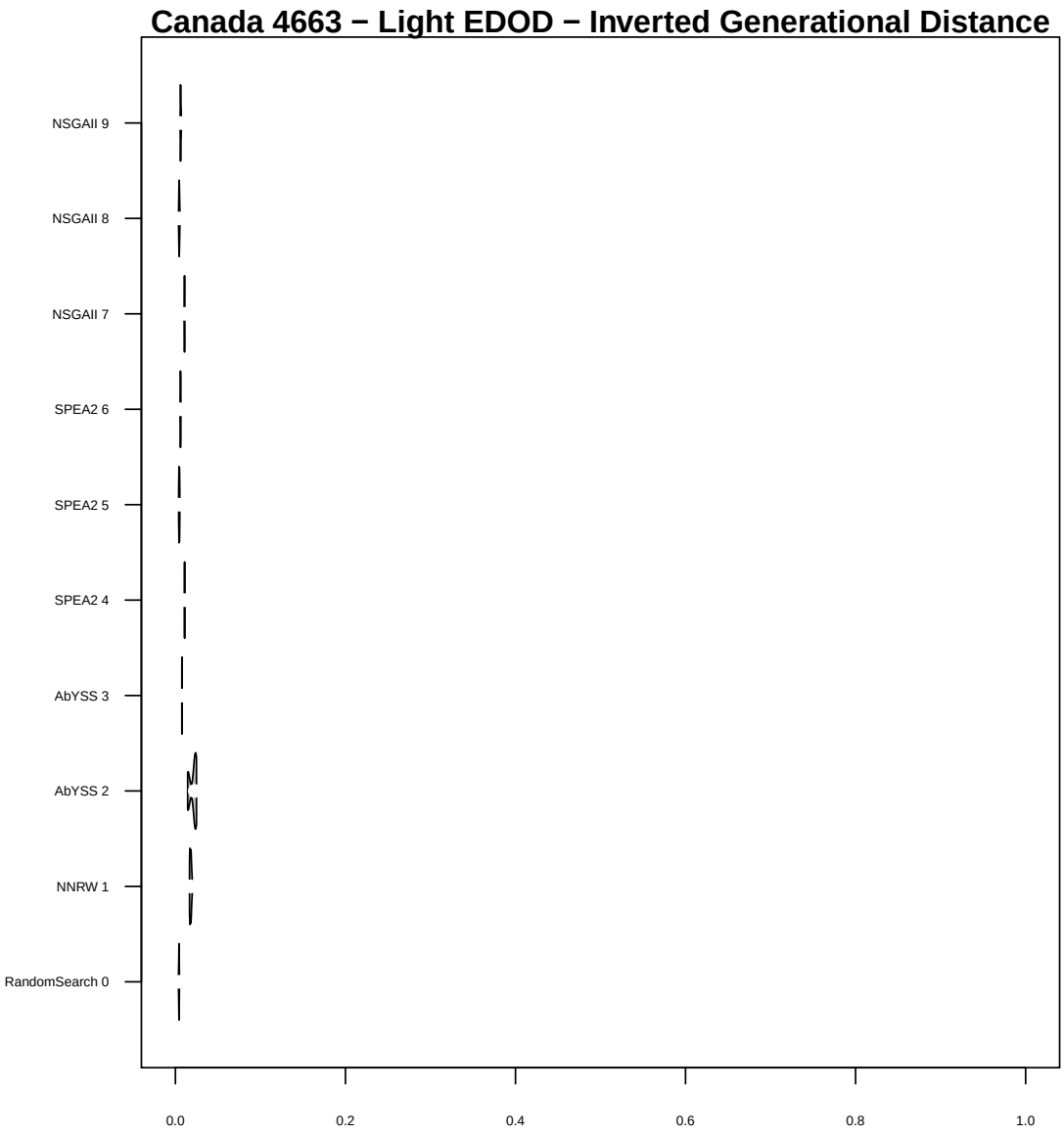



Canada 4663 − Light EDOD − Epsilon
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0e+00
1e+08
2e+08
3e+08
4e+08
5e+08


Canada 4663 − Light EDOD − Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

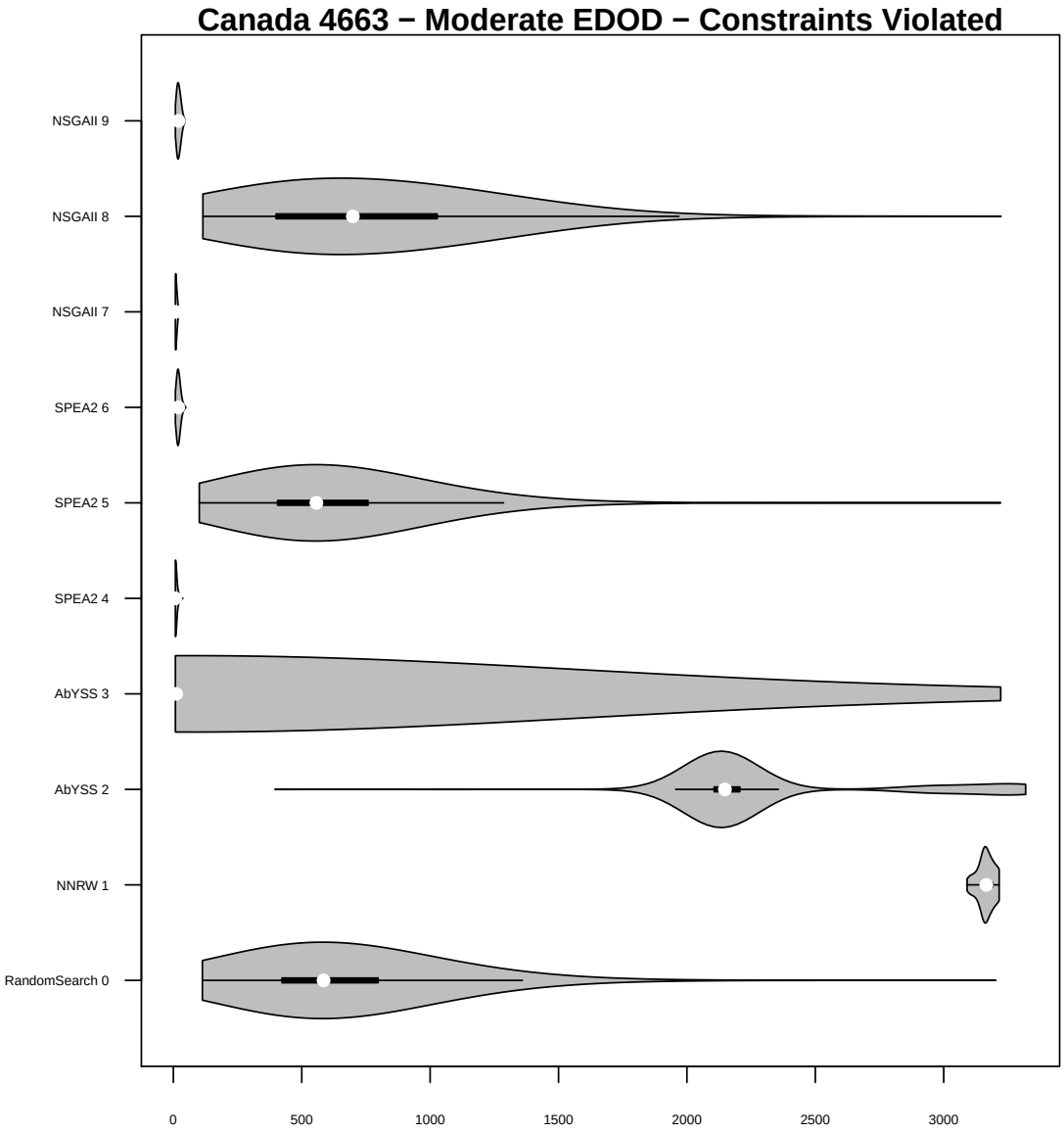

Canada 4663 − Light EDOD − Generalized Spread
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

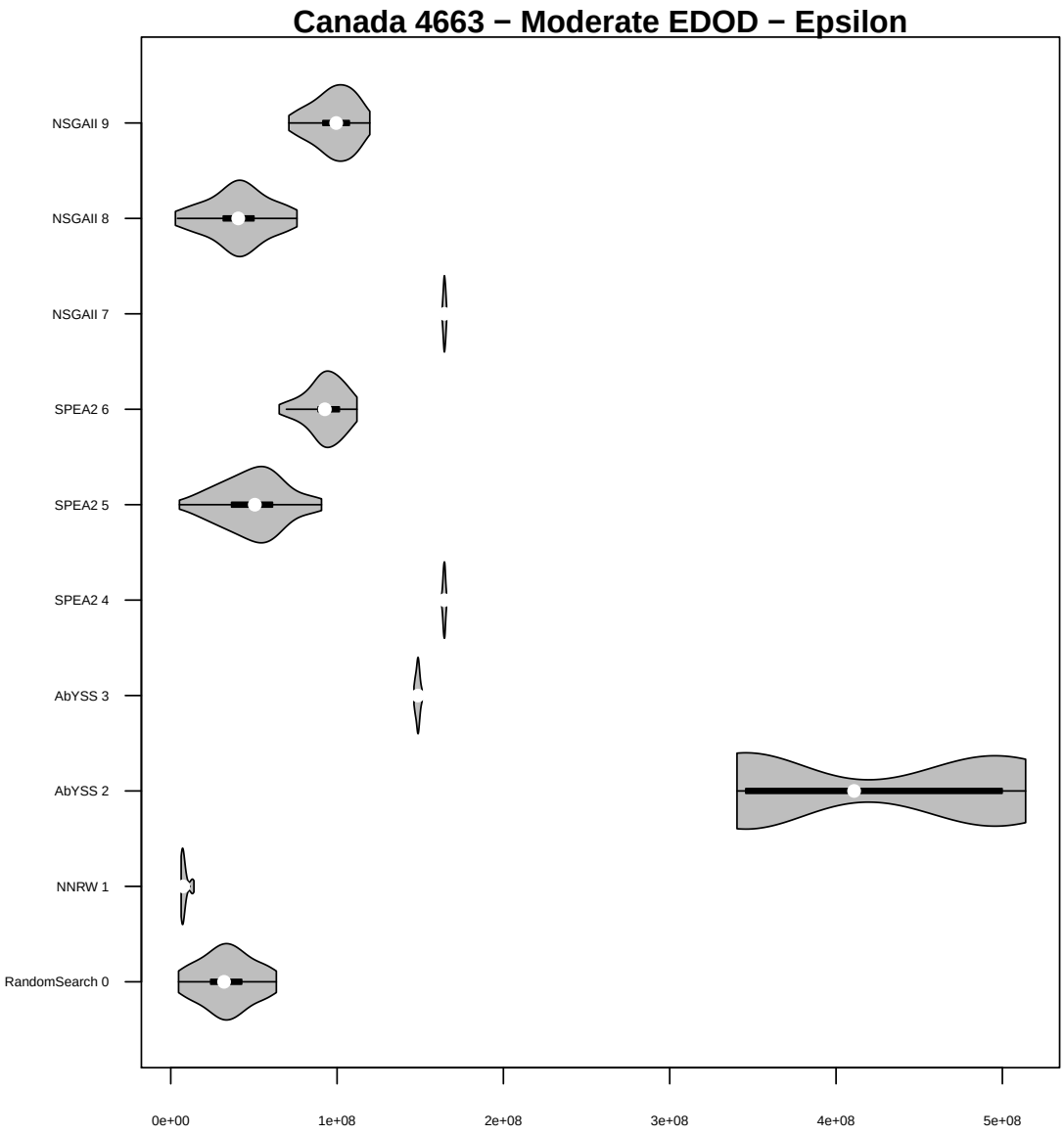

Canada 4663 − Light EDOD − Hypervolume
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

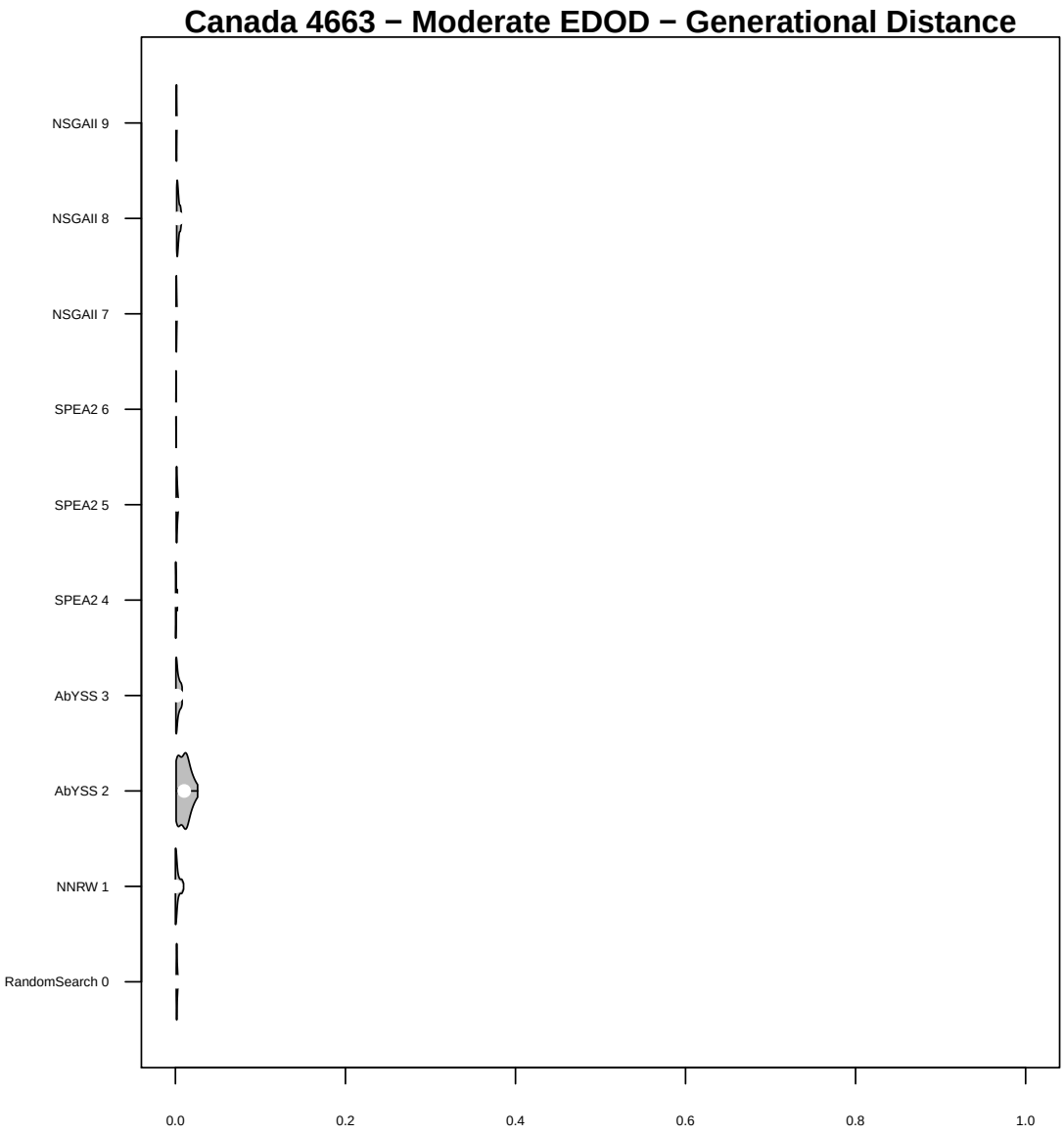

Canada 4663 − Light EDOD − Inverted Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

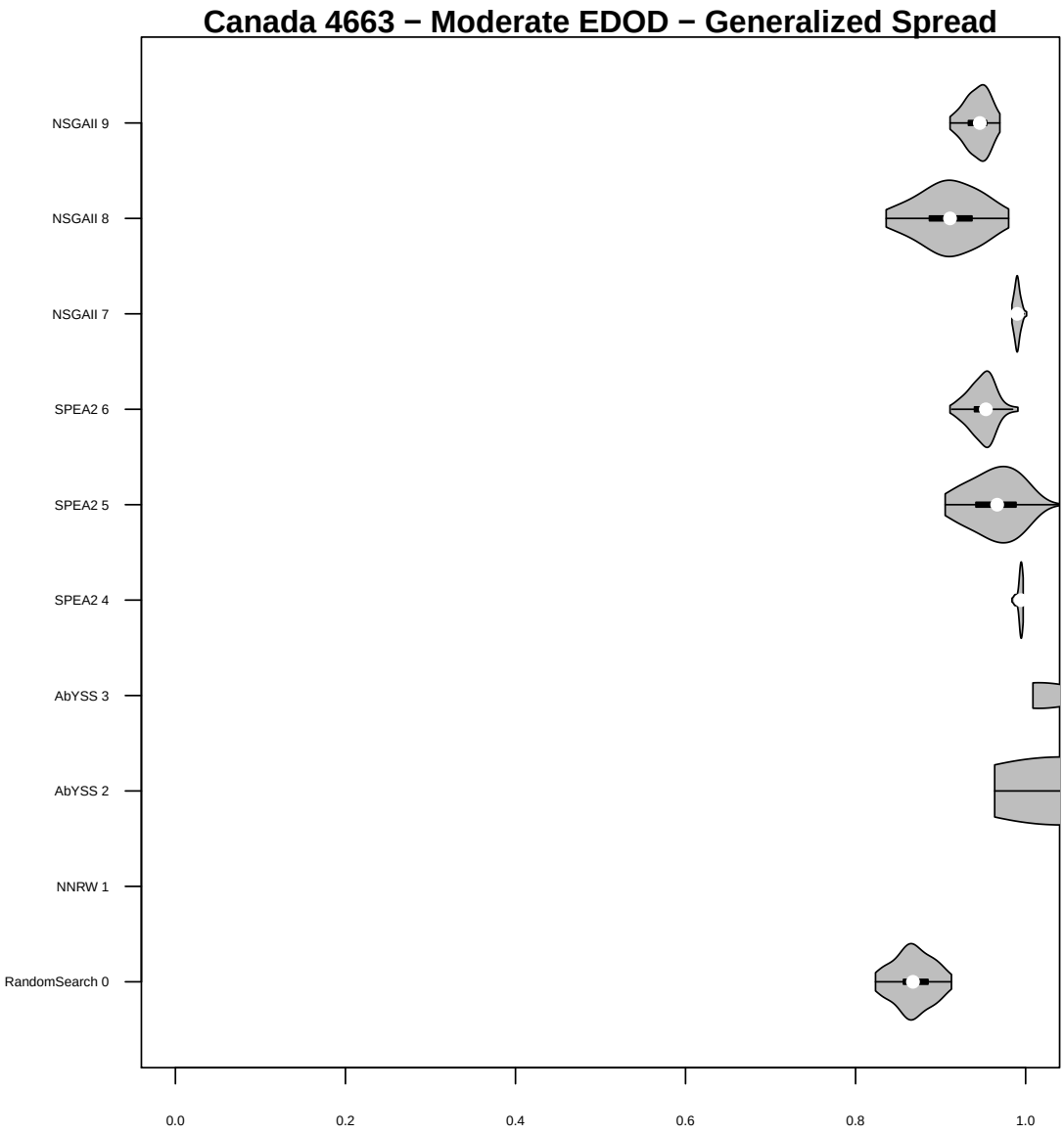
A.3 Canada

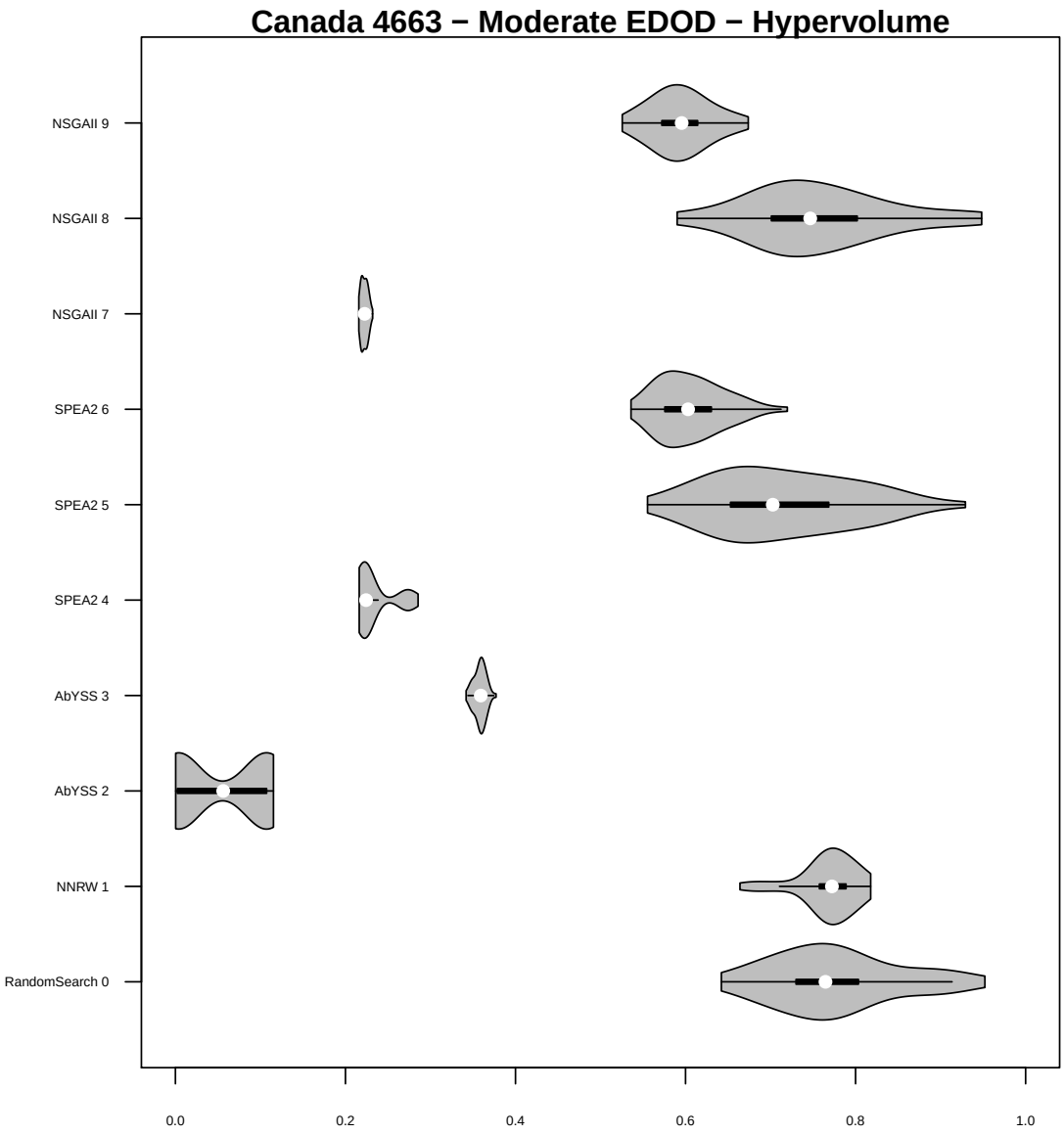
A.3.1 Moderate EDOD.

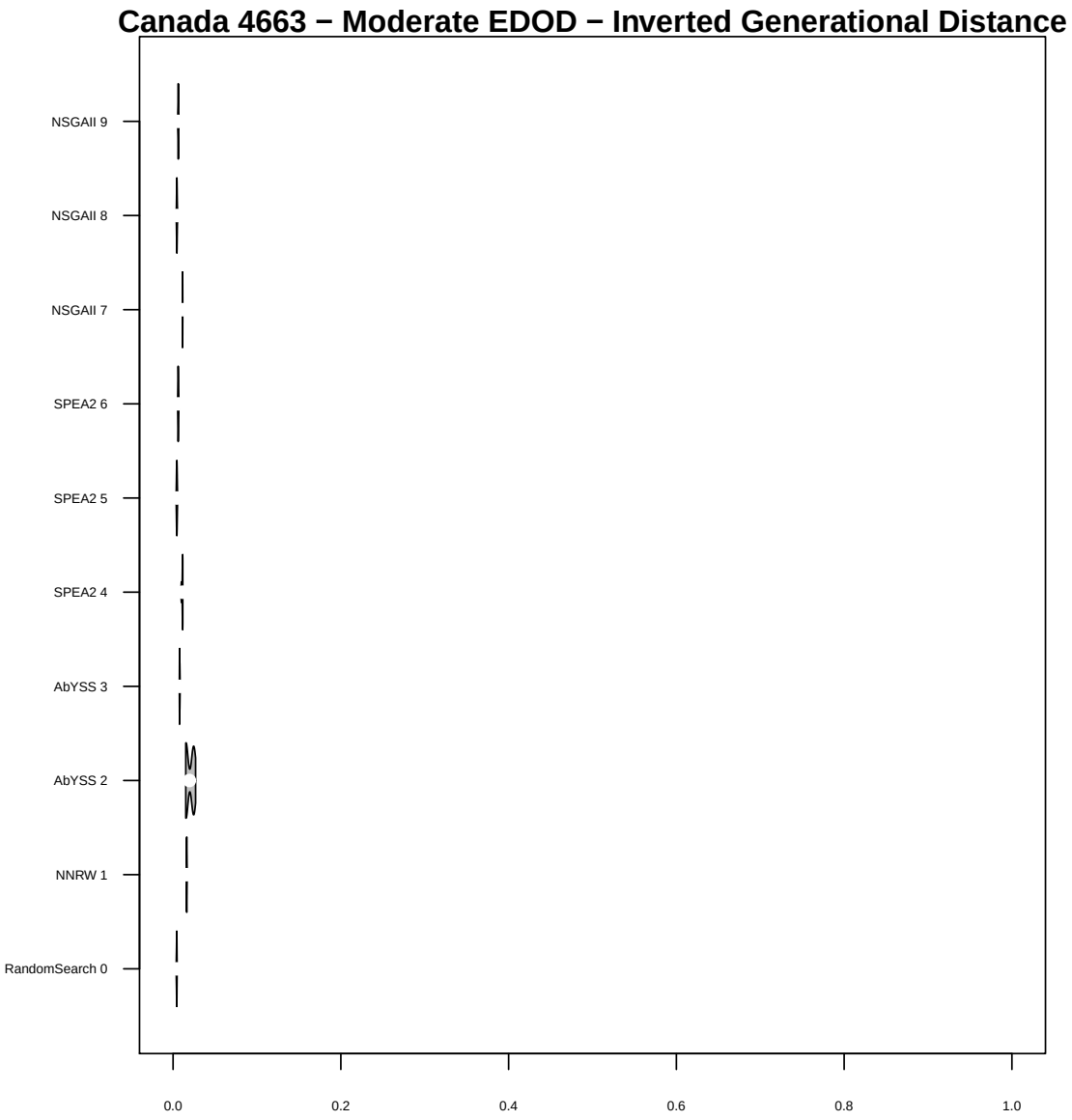



Canada 4663 − Moderate EDOD − Epsilon
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0e+00
1e+08
2e+08
3e+08
4e+08
5e+08


Canada 4663 − Moderate EDOD − Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

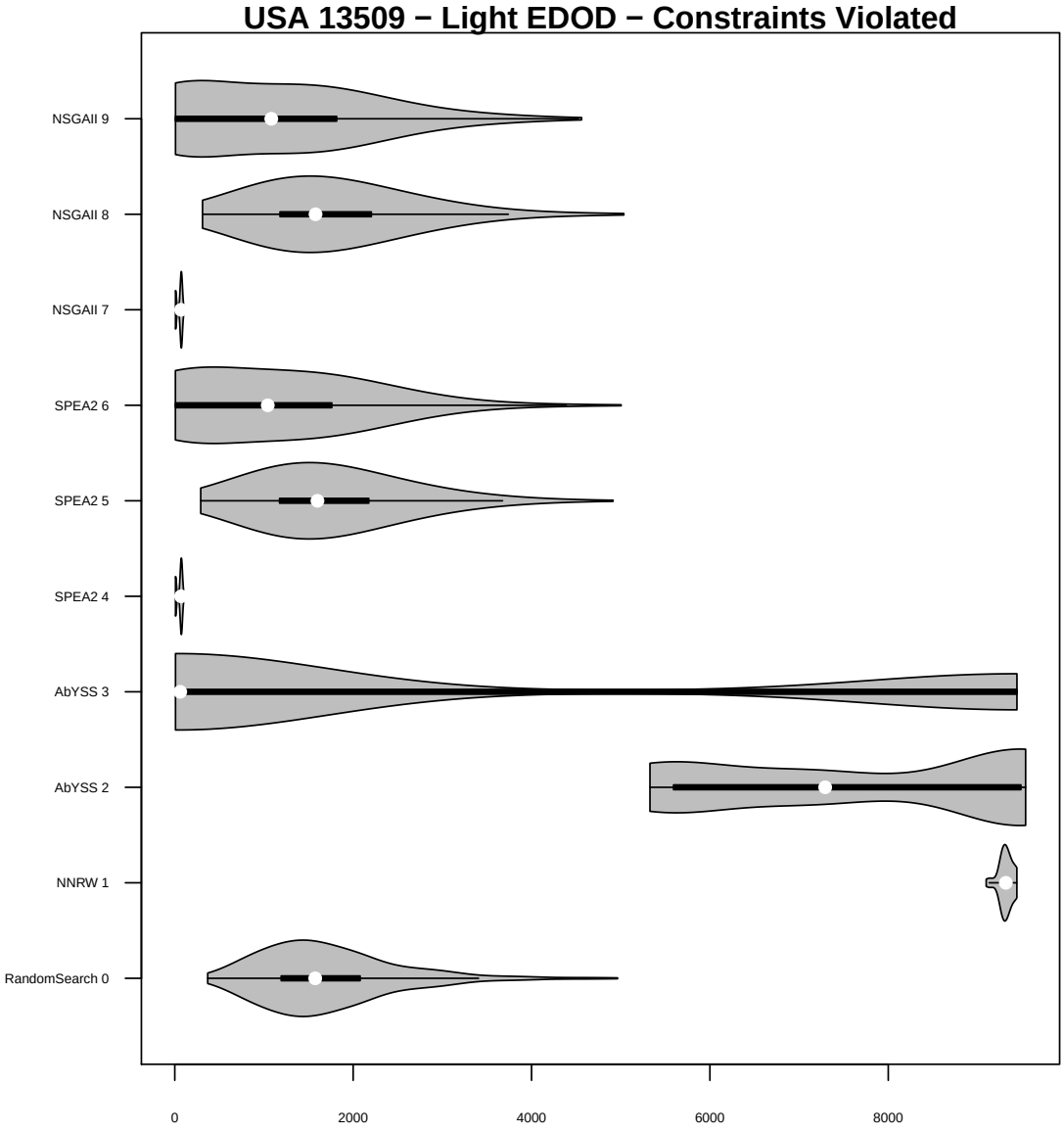

Canada 4663 − Moderate EDOD − Generalized Spread
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

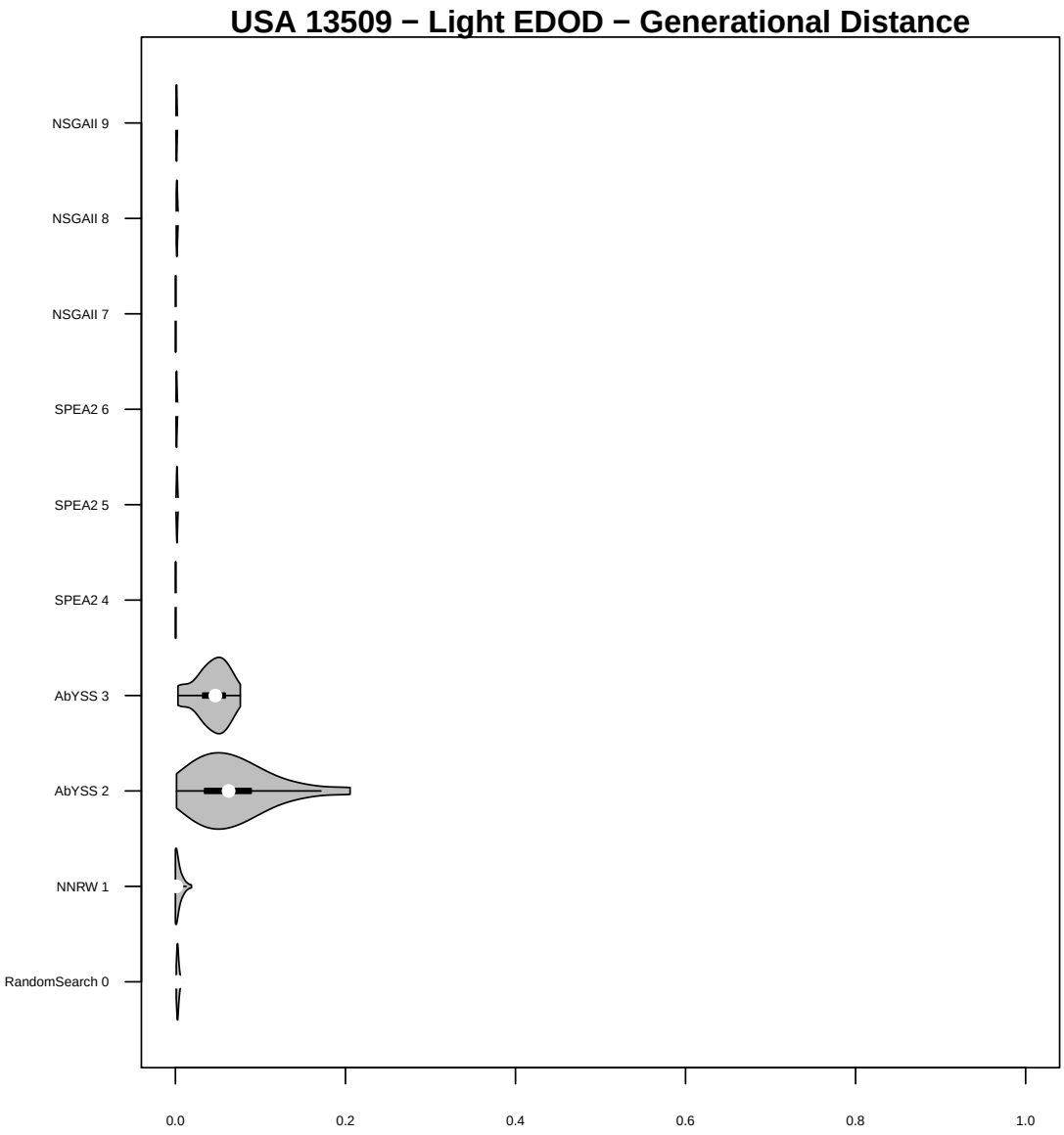

Canada 4663 − Moderate EDOD − Hypervolume
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

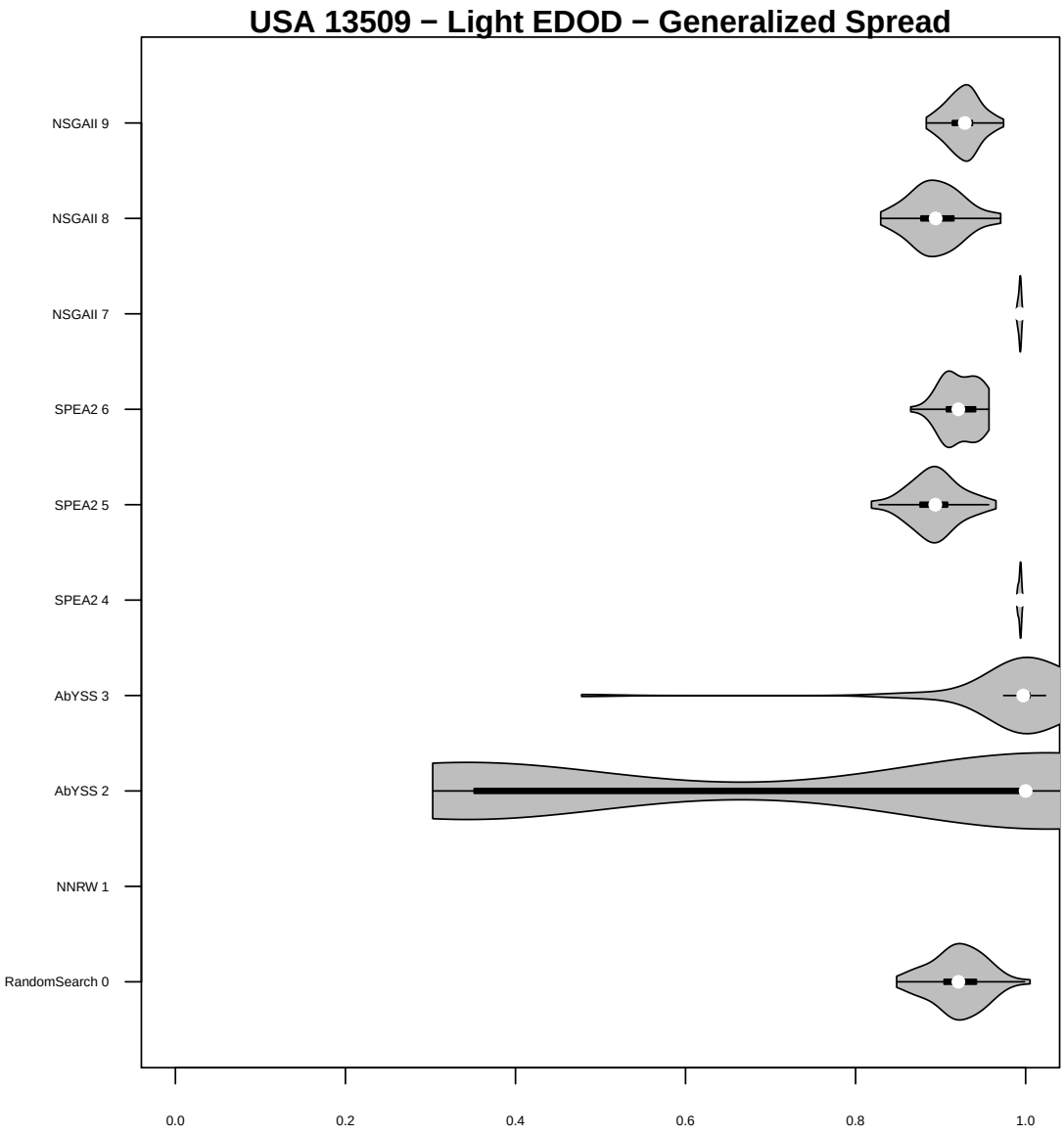

Canada 4663 − Moderate EDOD − Inverted Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

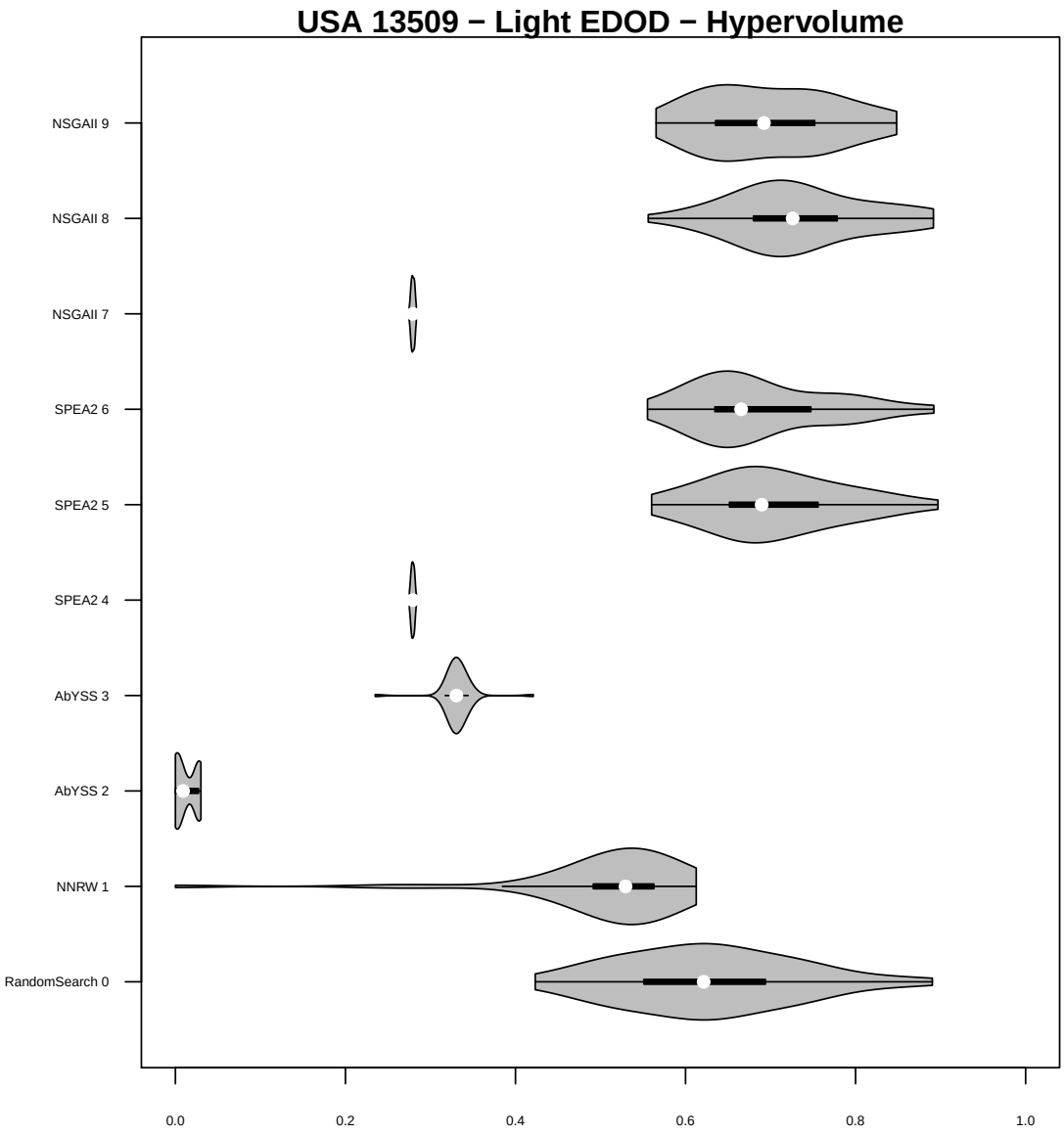
A.4 USA

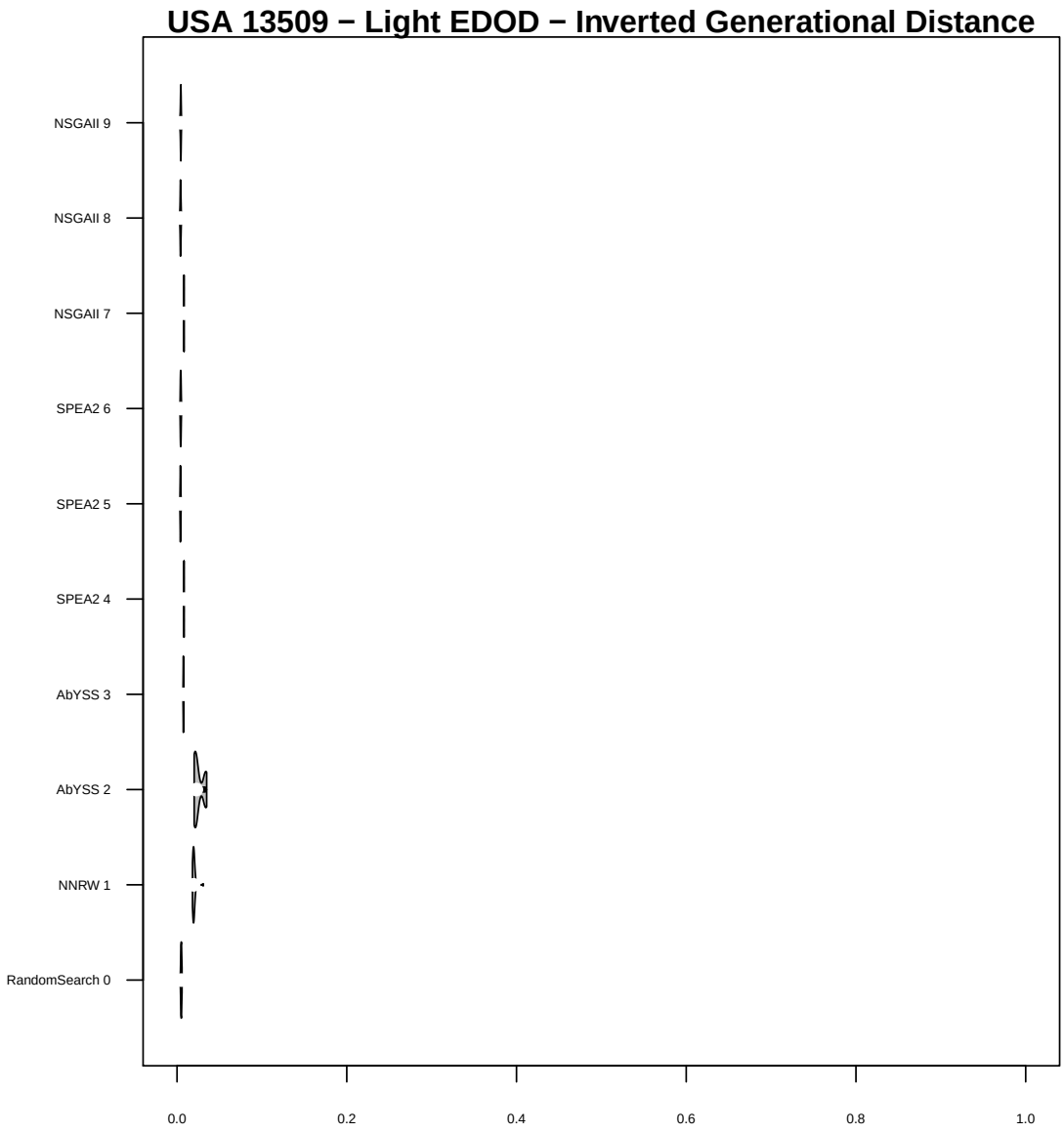
A.4.1 Light EDOD.



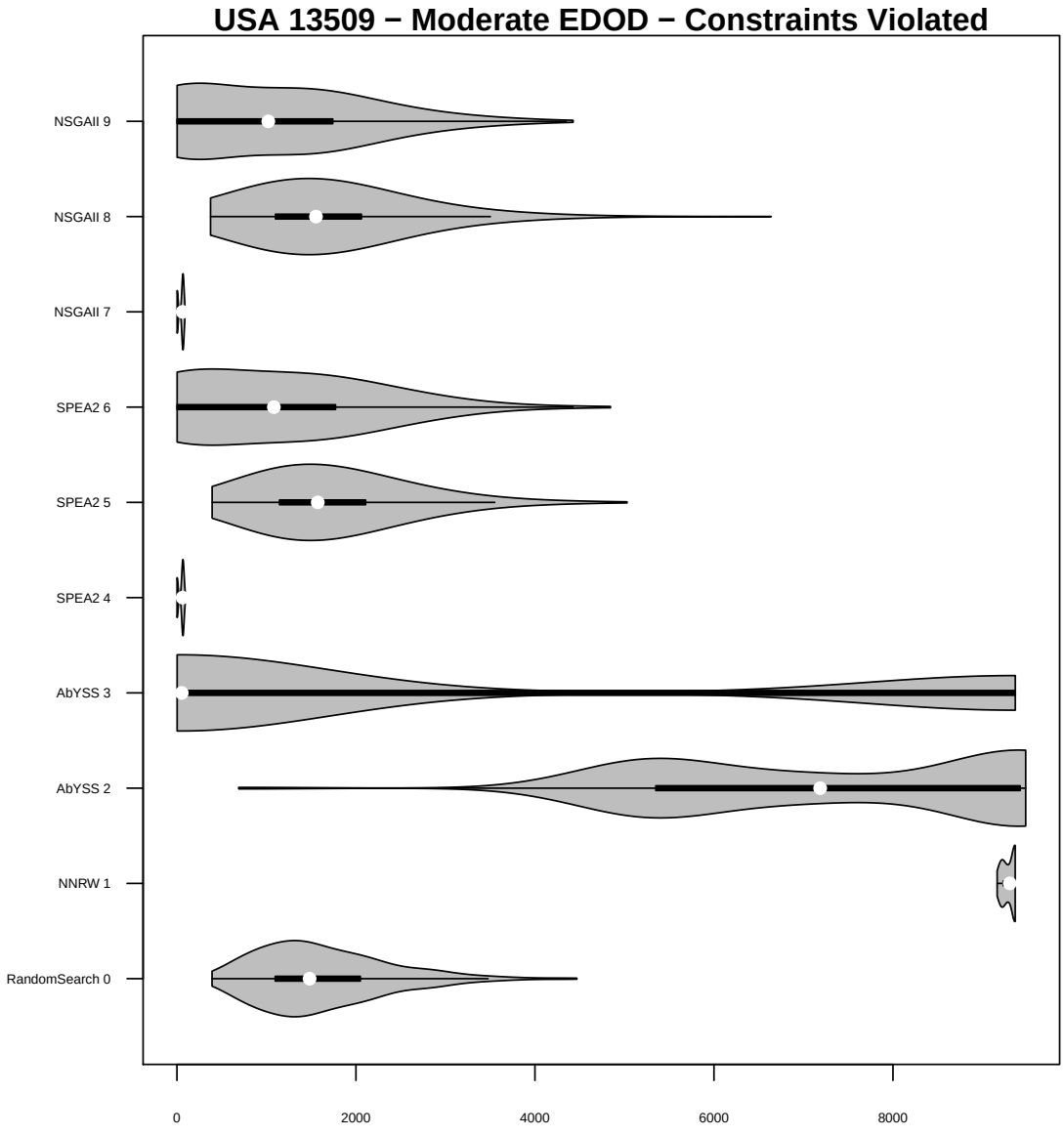

USA 13509 − Light EDOD − Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

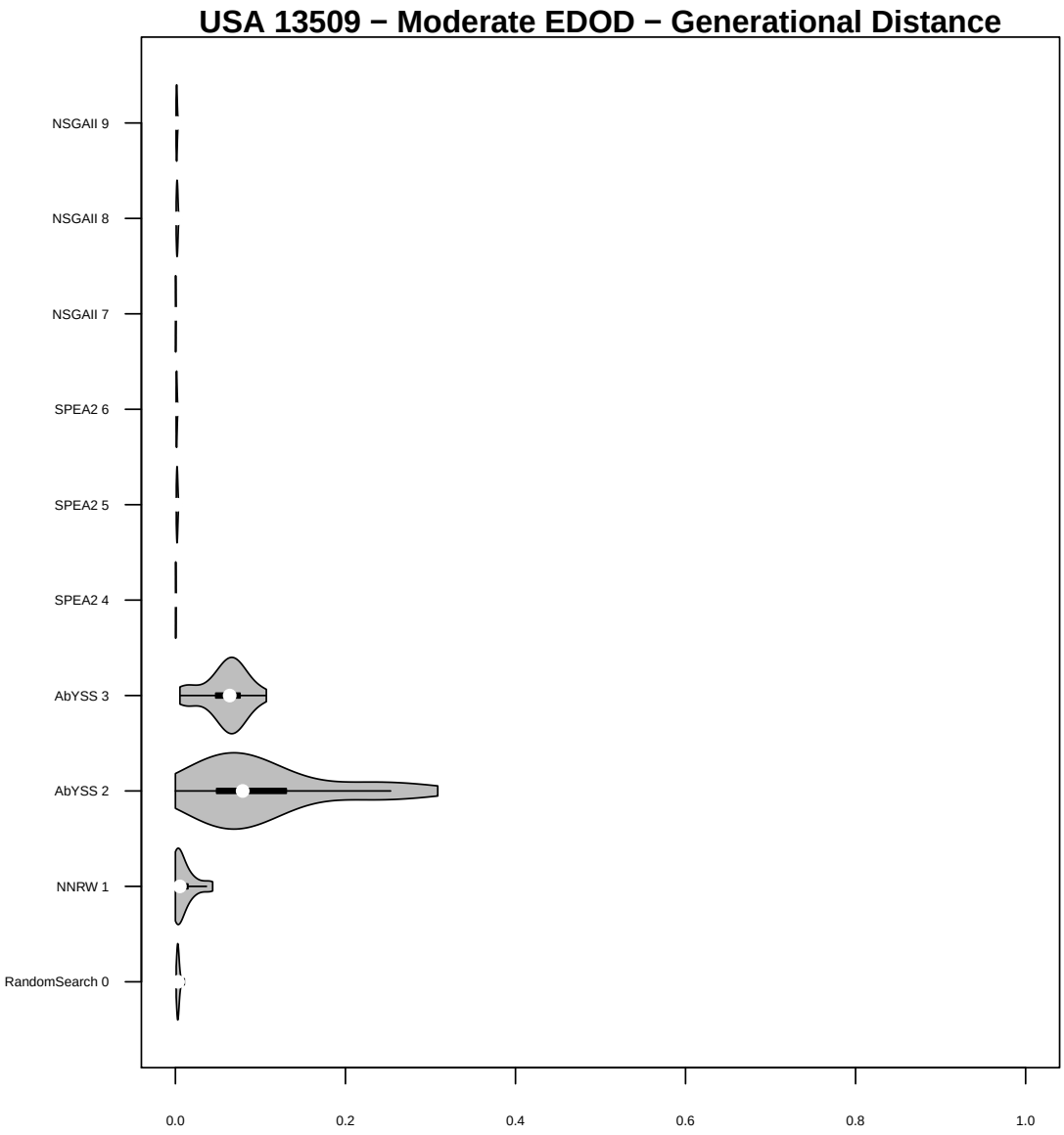

USA 13509 − Light EDOD − Generalized Spread
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

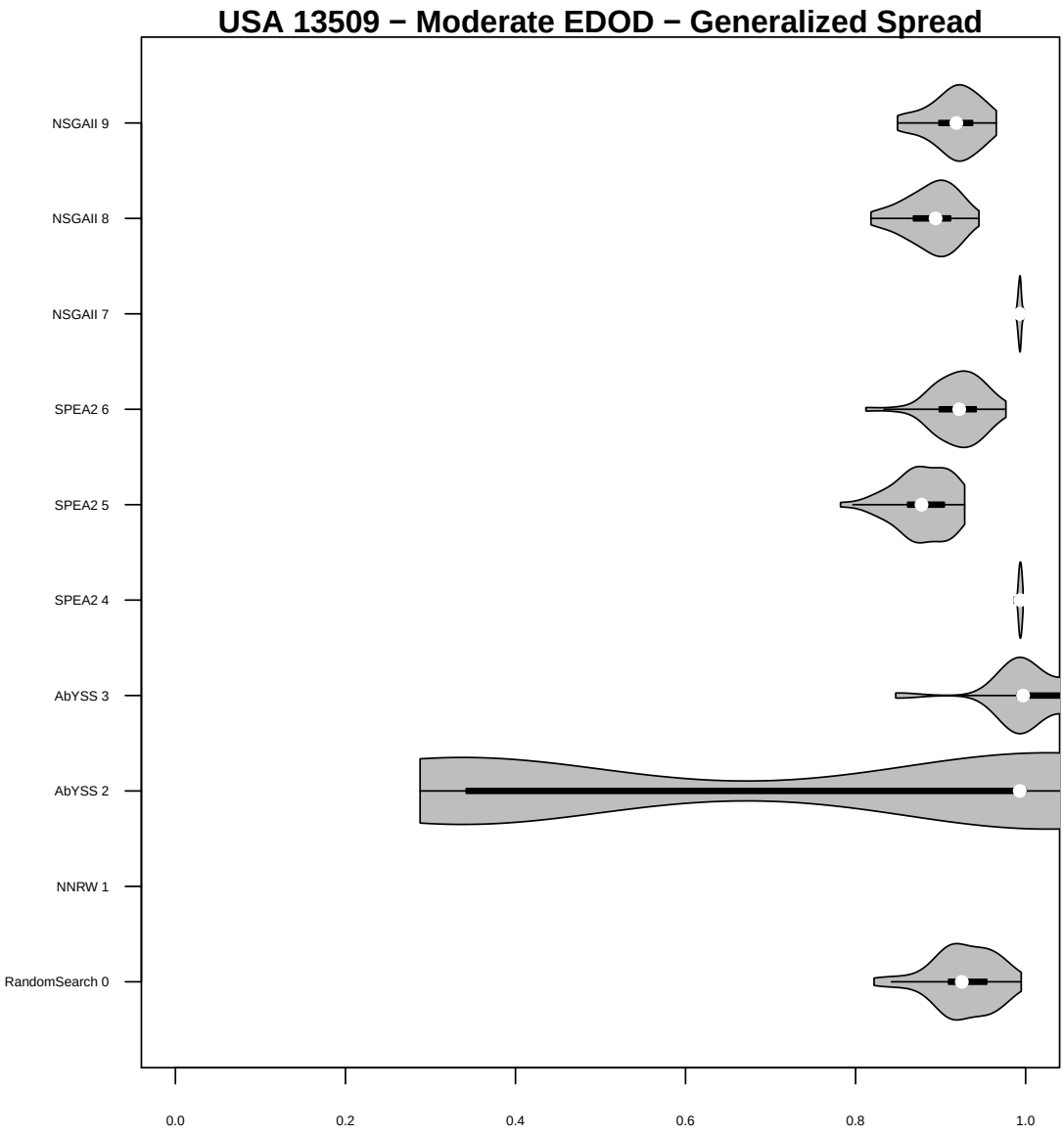

USA 13509 − Light EDOD − Hypervolume
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

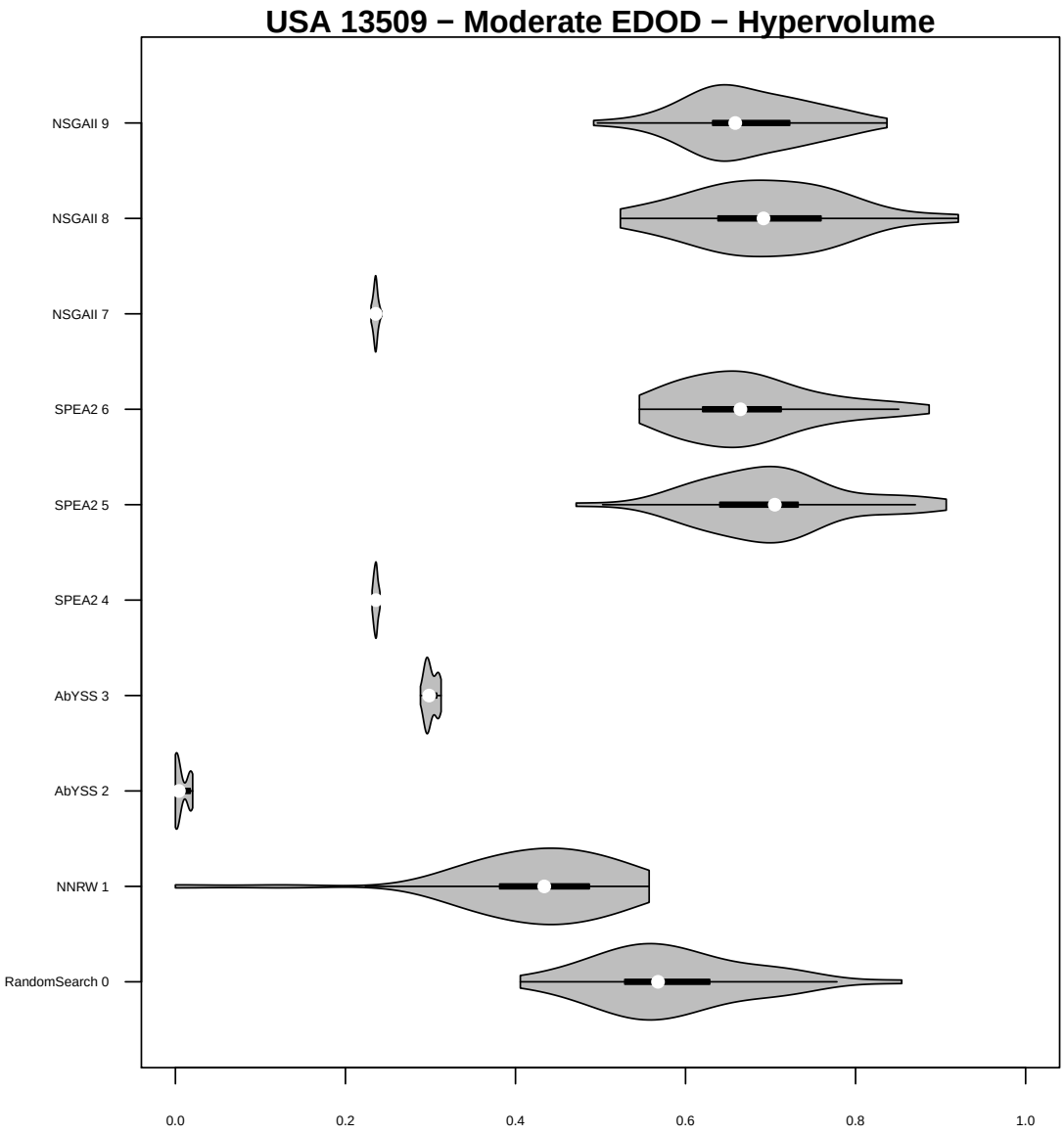

USA 13509 − Light EDOD − Inverted Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

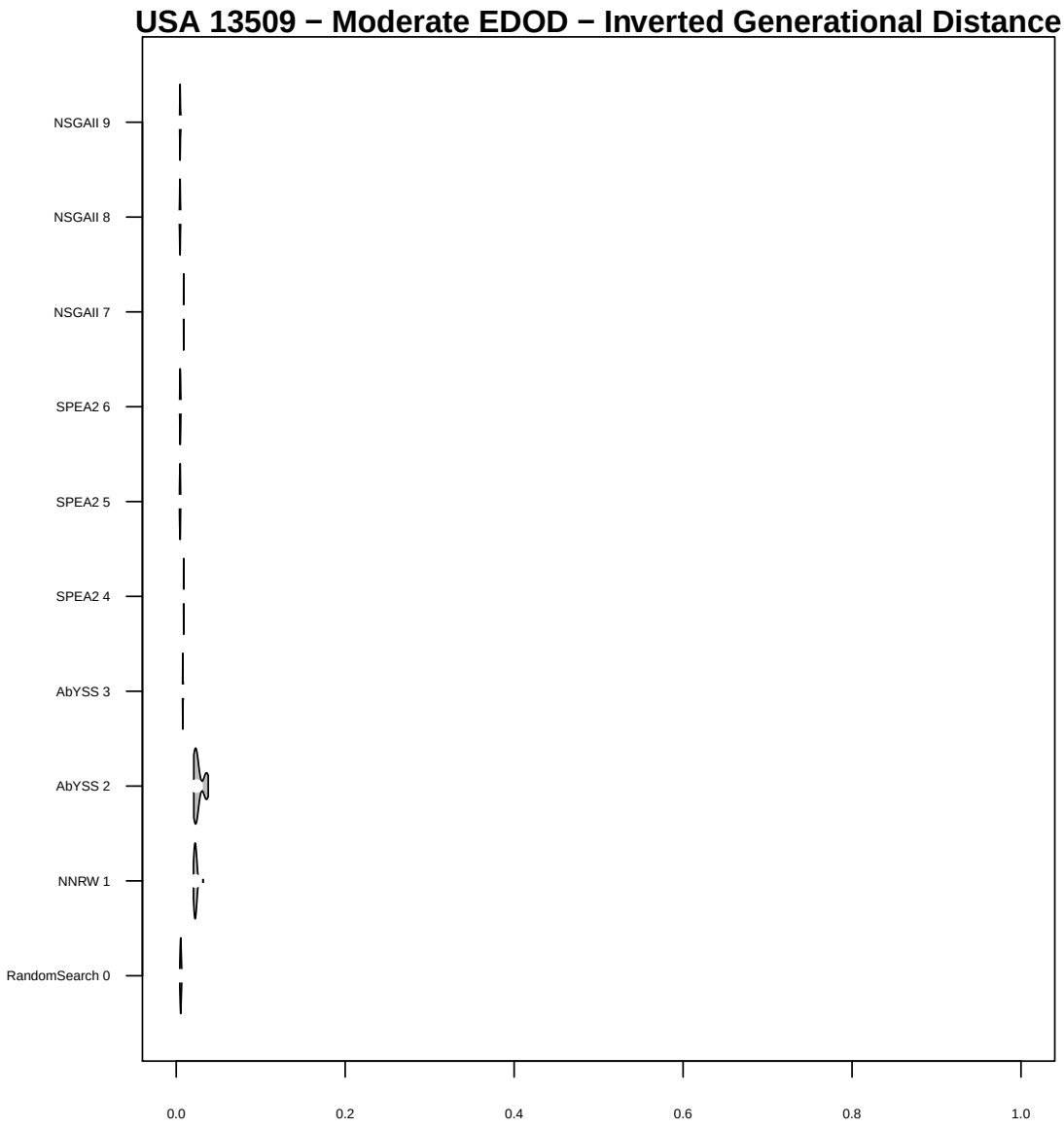
A.4.2 Moderate EDOD.




USA 13509 − Moderate EDOD − Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0


USA 13509 − Moderate EDOD − Generalized Spread
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0


USA 13509 − Moderate EDOD − Hypervolume
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

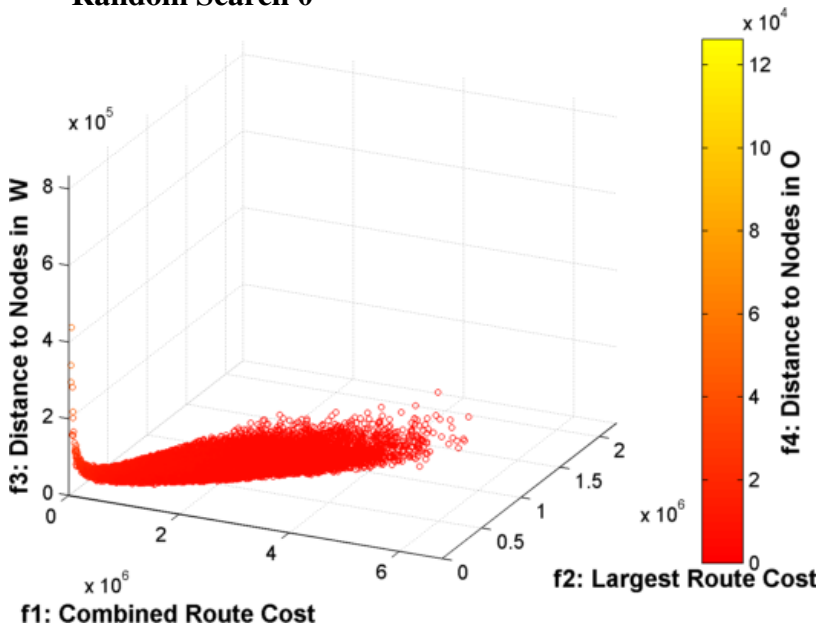

USA 13509 − Moderate EDOD − Inverted Generational Distance
NSGAII 9
NSGAII 8
NSGAII 7
SPEA2 6
SPEA2 5
SPEA2 4
AbYSS 3
AbYSS 2
NNRW 1
RandomSearch 0
0.0
0.2
0.4
0.6
0.8
1.0

Appendix B: Pareto Front Plots

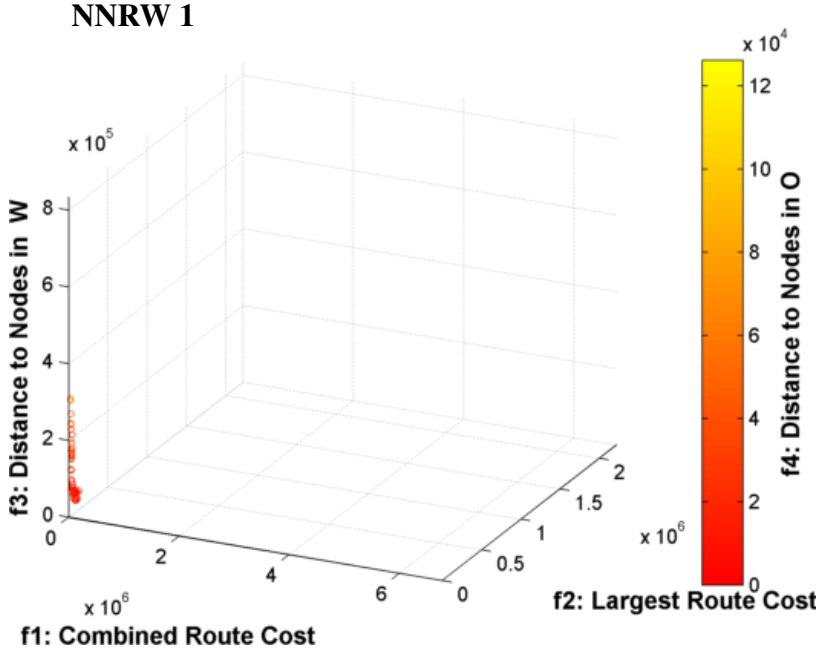
B.1 Uruguay

B.1.1 Light EDOD.

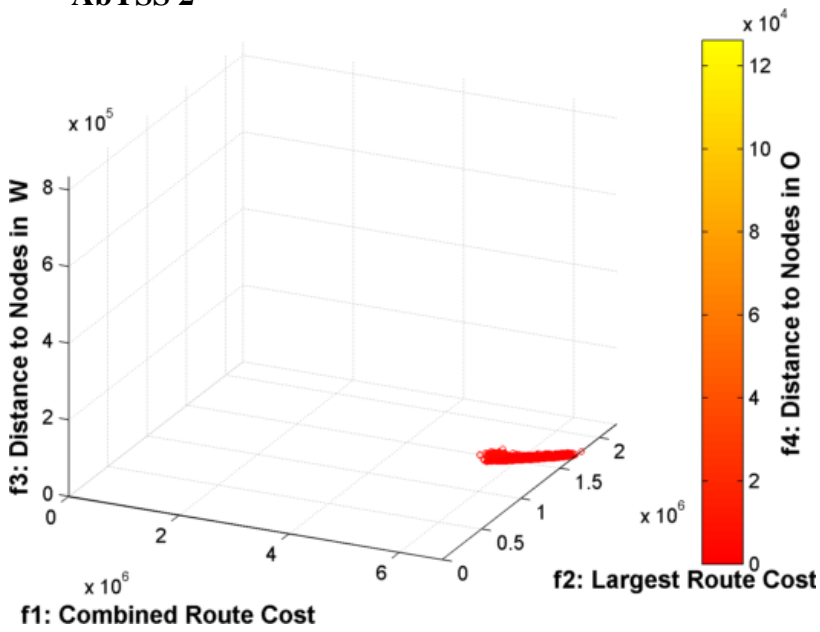
Random Search 0



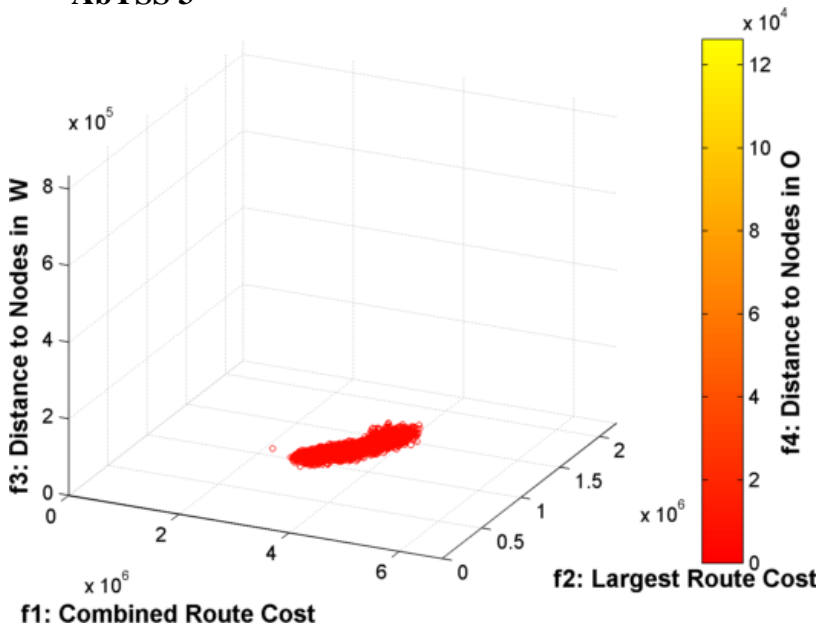
NNRW 1



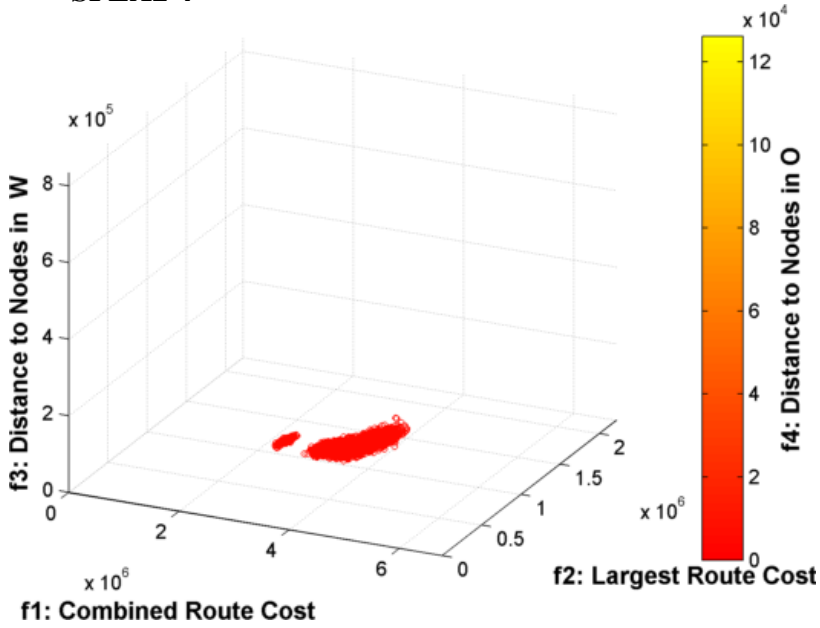
AbYSS 2



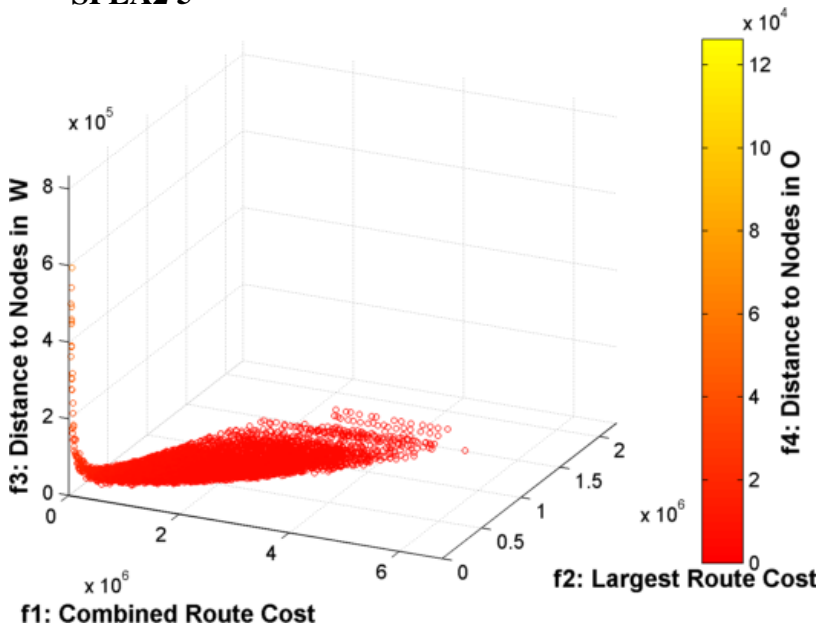
AbYSS 3



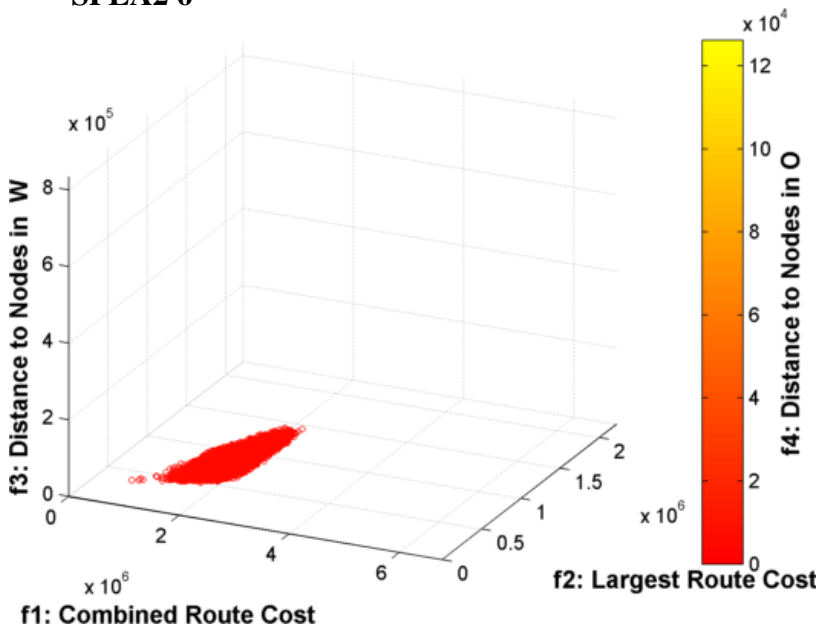
SPEA2 4



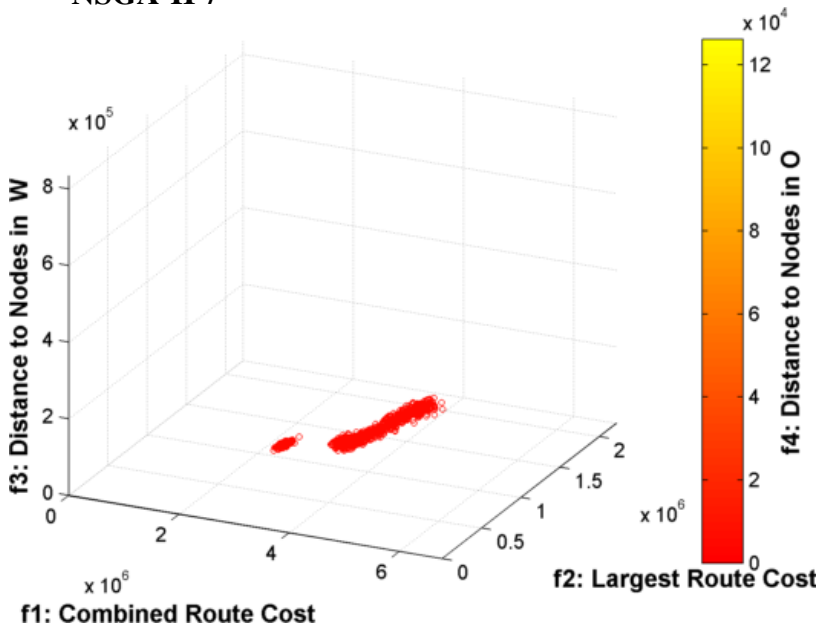
SPEA2 5



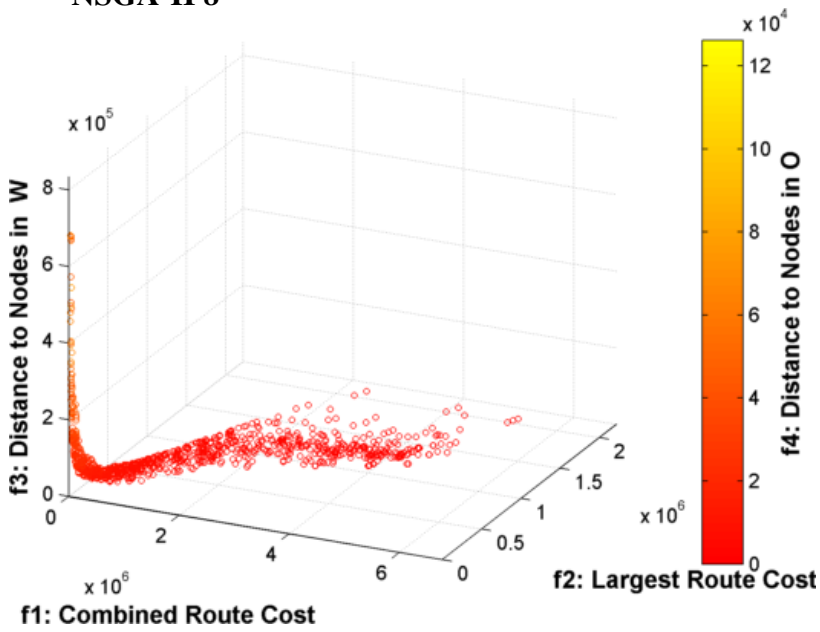
SPEA2 6



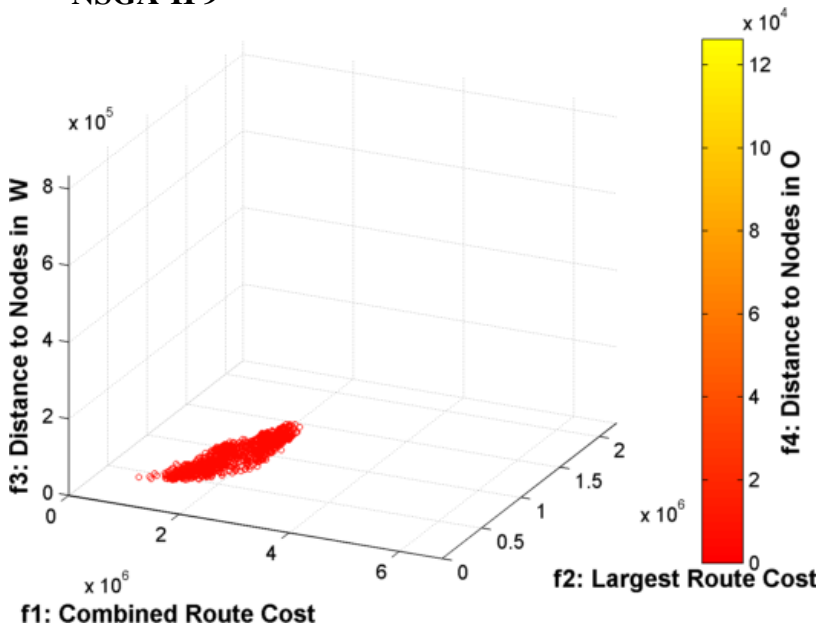
NSGA-II 7



NSGA-II 8

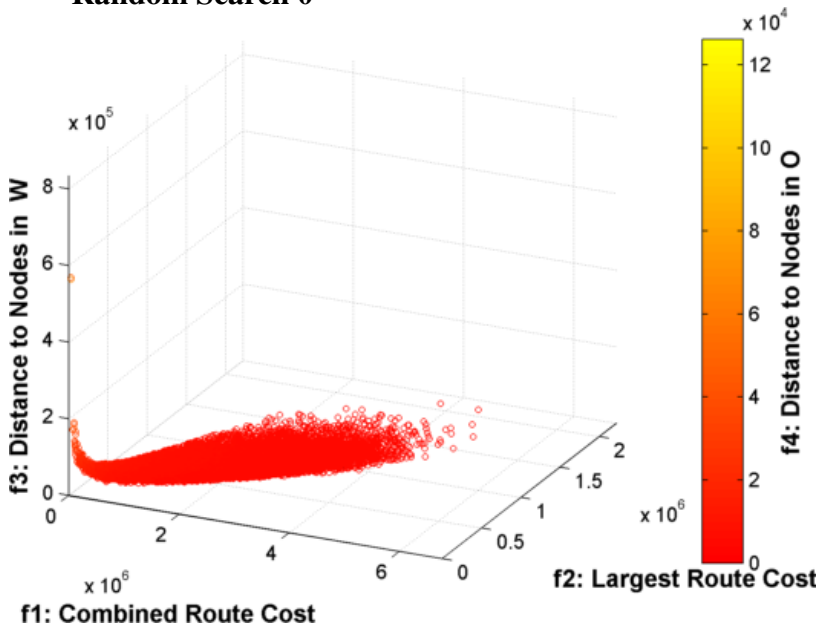


NSGA-II 9

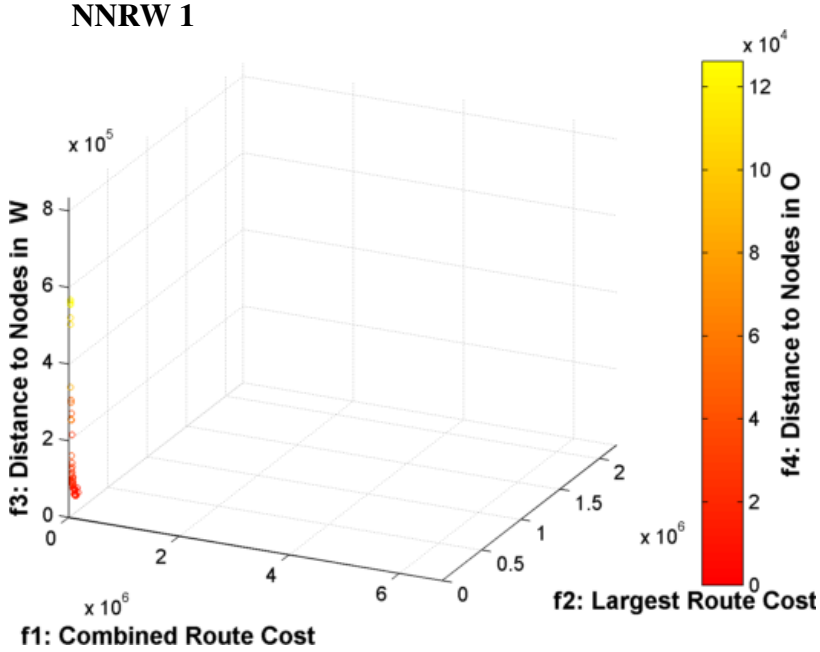


B.1.2 Moderate EDOD.

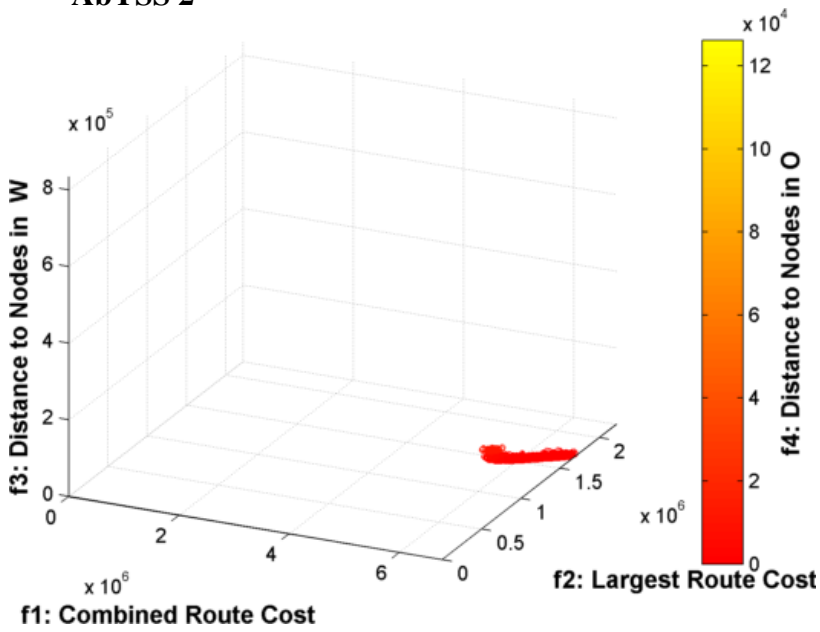
Random Search 0



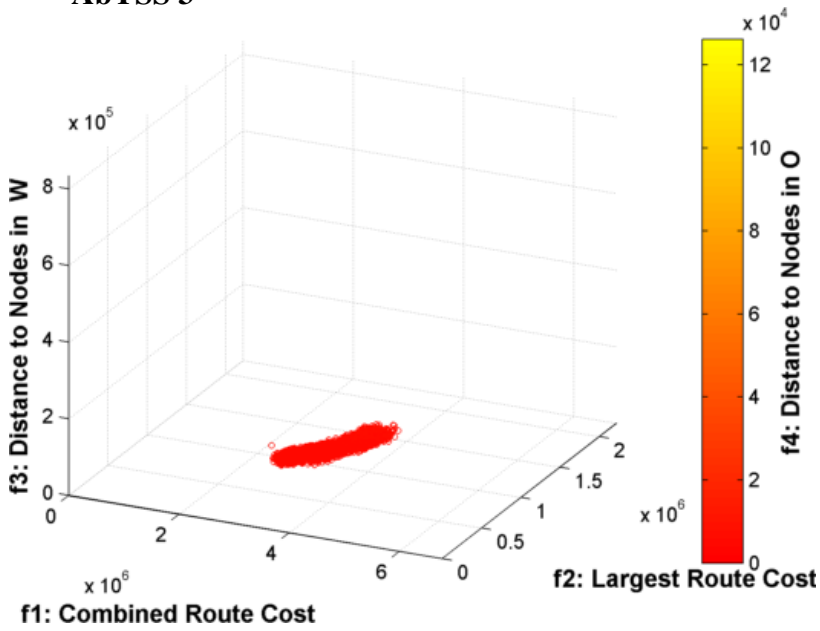
NNRW 1



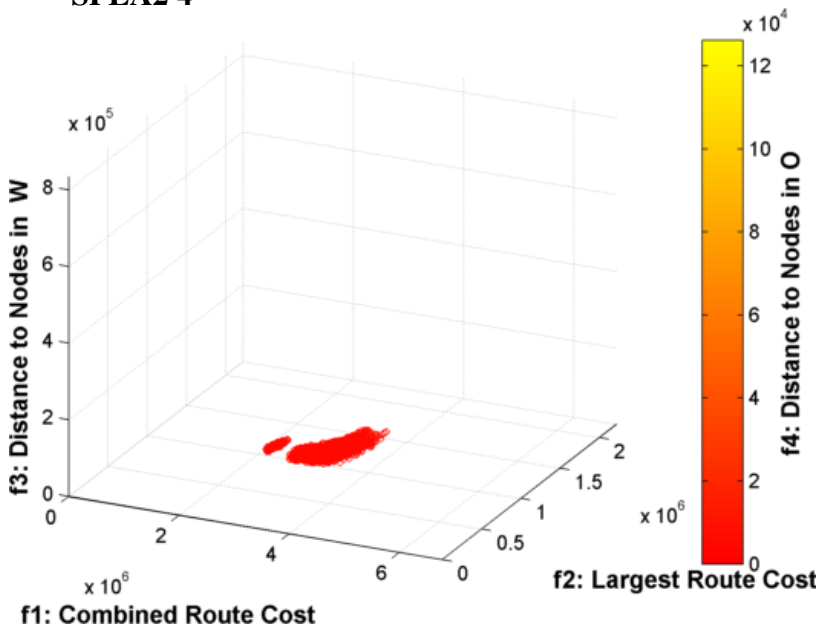
AbYSS 2



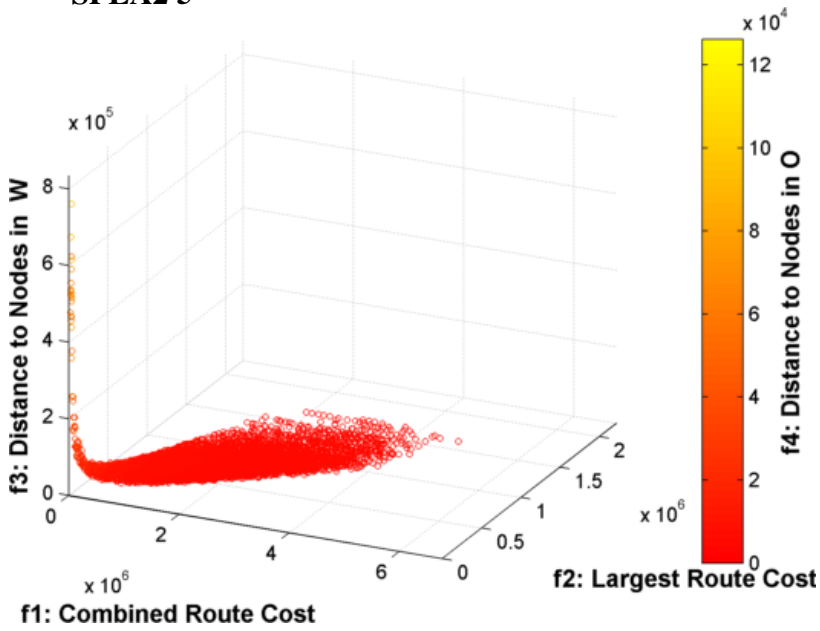
AbYSS 3



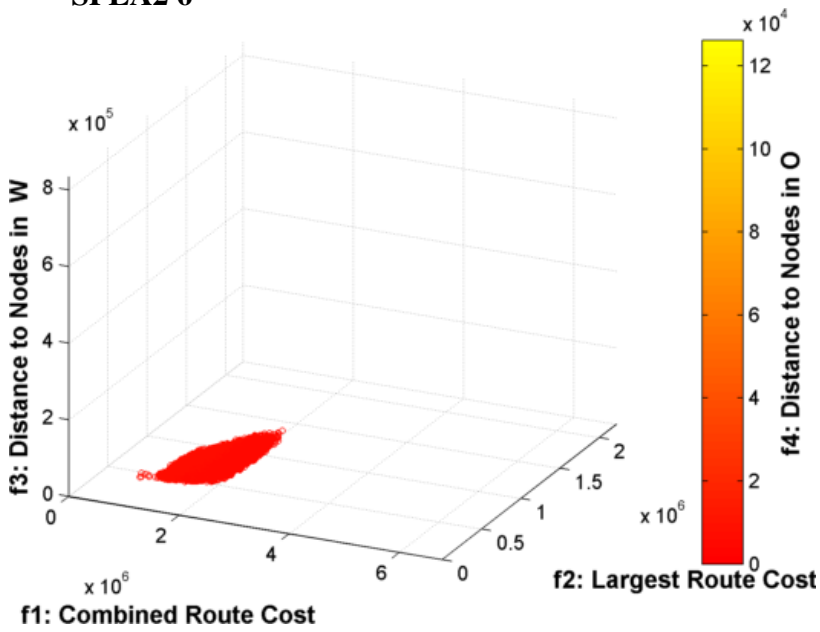
SPEA2 4



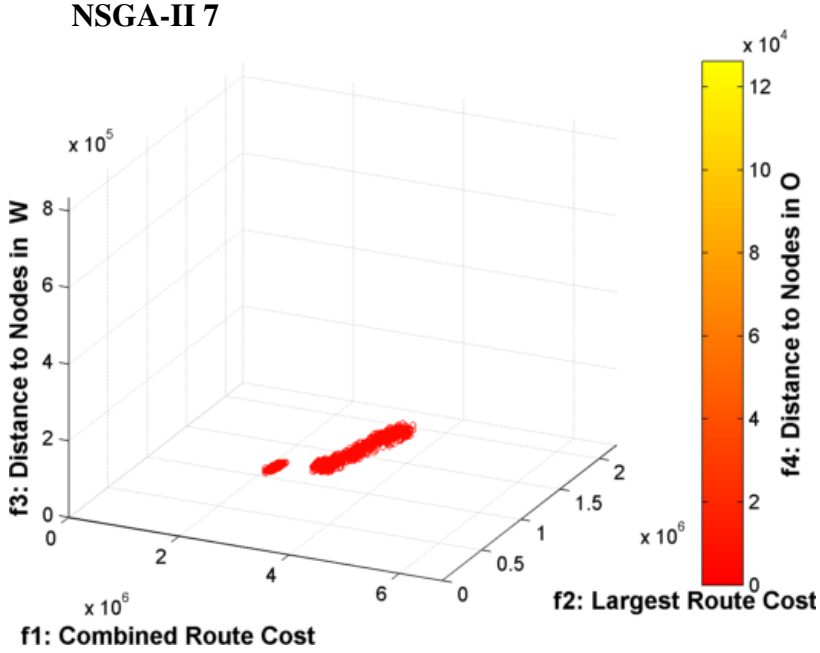
SPEA2 5



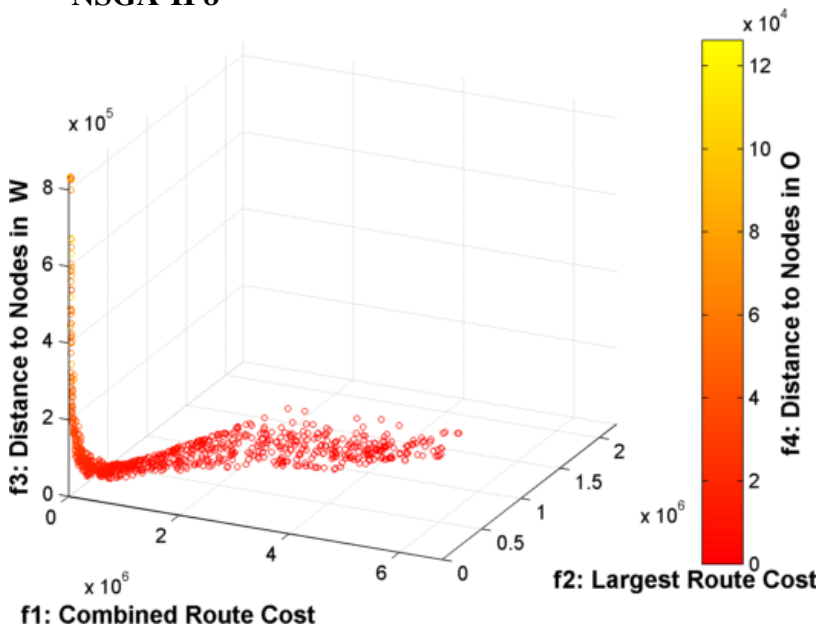
SPEA2 6



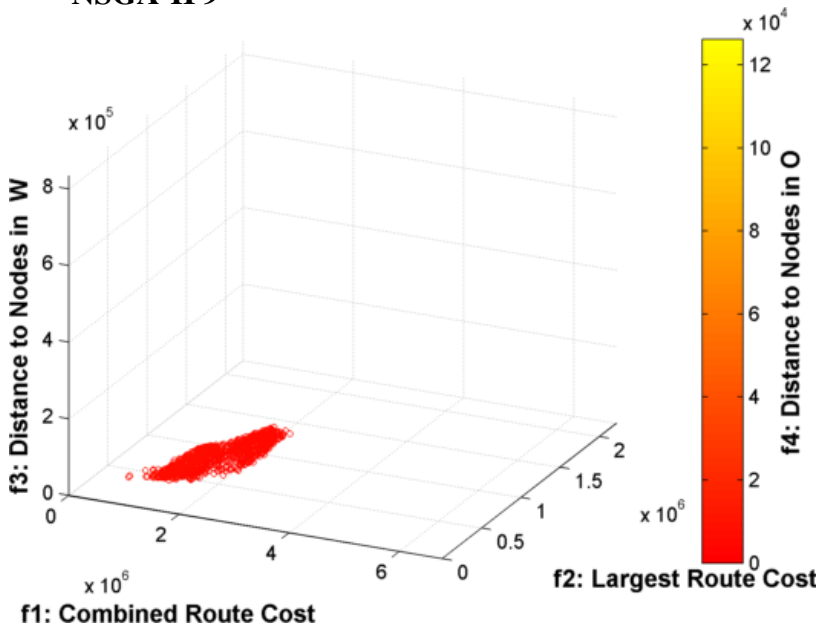
NSGA-II 7



NSGA-II 8



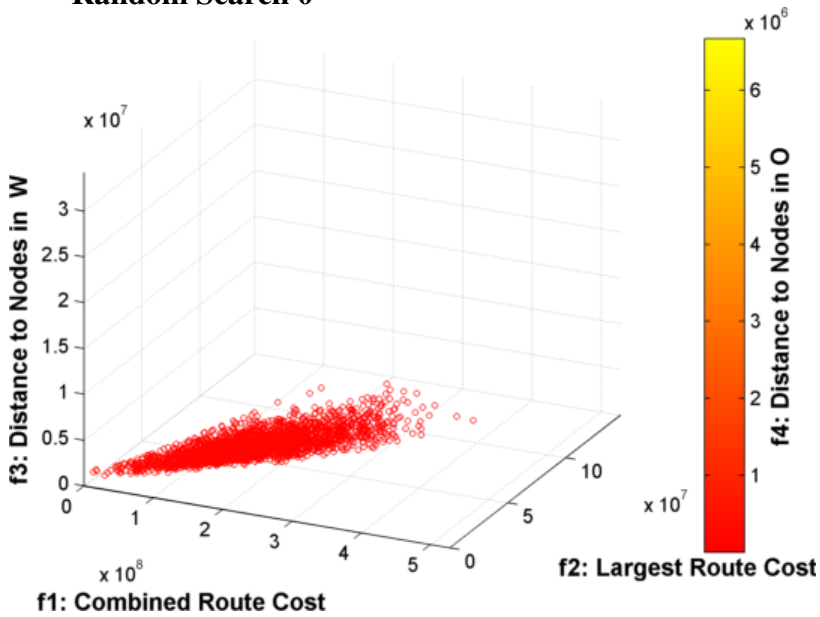
NSGA-II 9



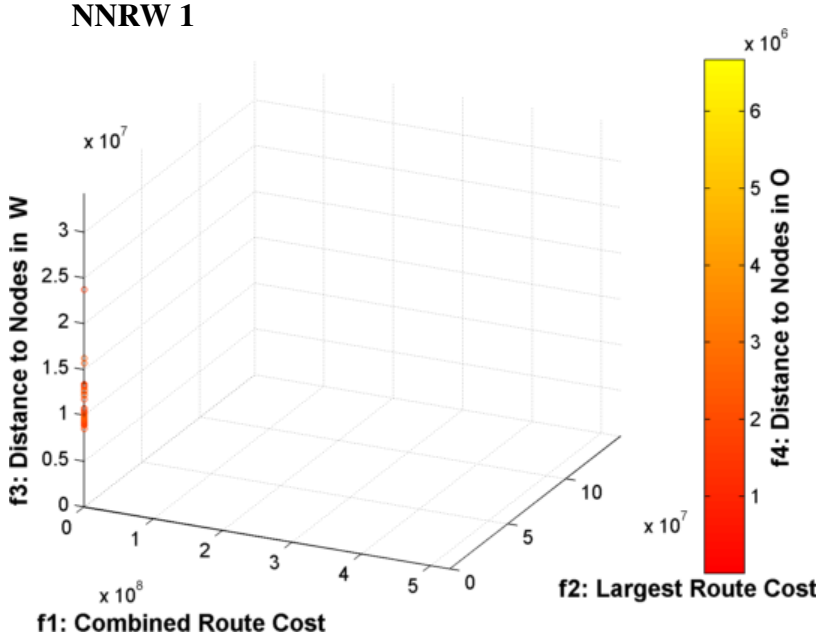
B.2 Canada

B.2.1 Light EDOD.

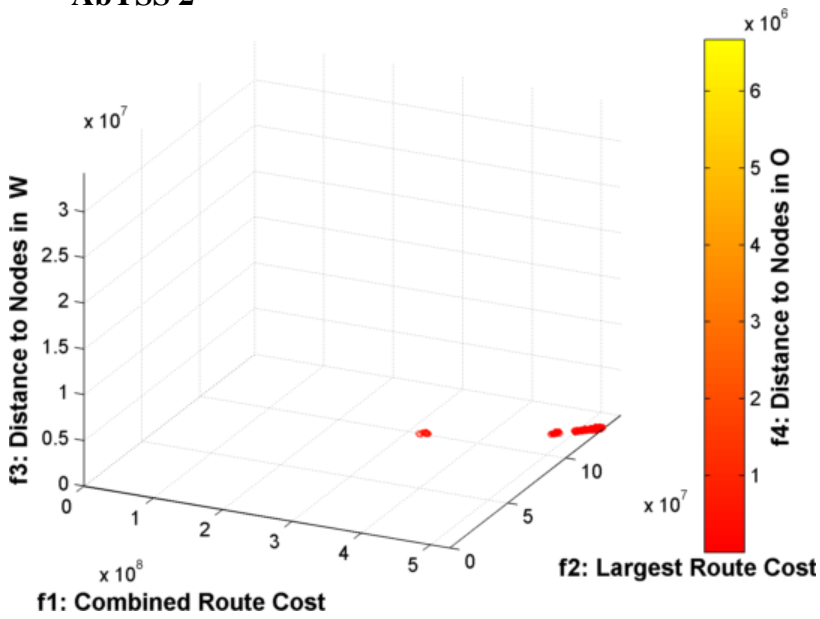
Random Search 0



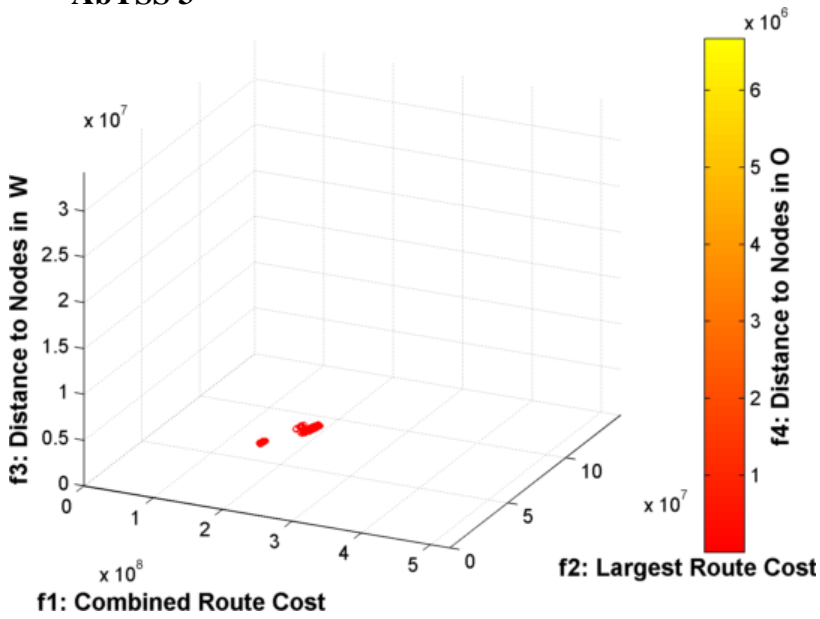
NNRW 1



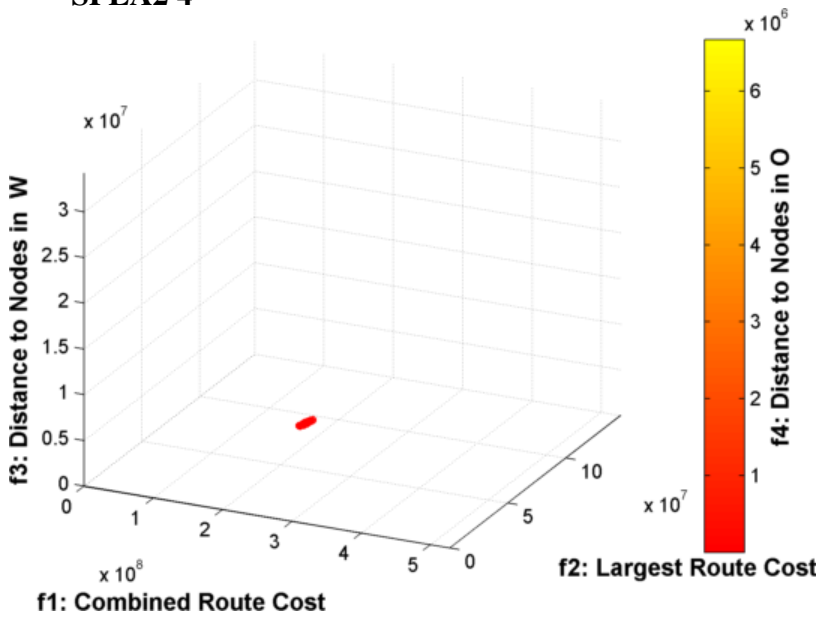
AbYSS 2



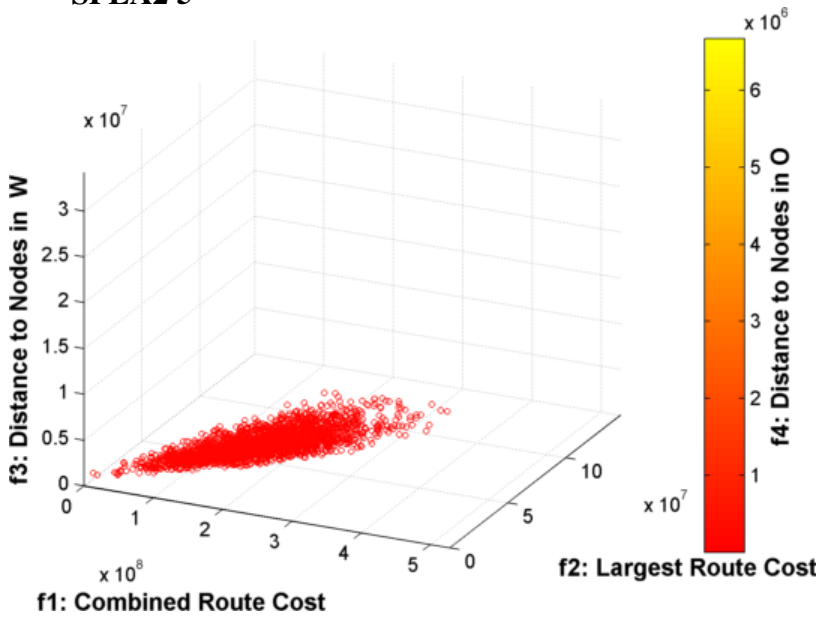
AbYSS 3



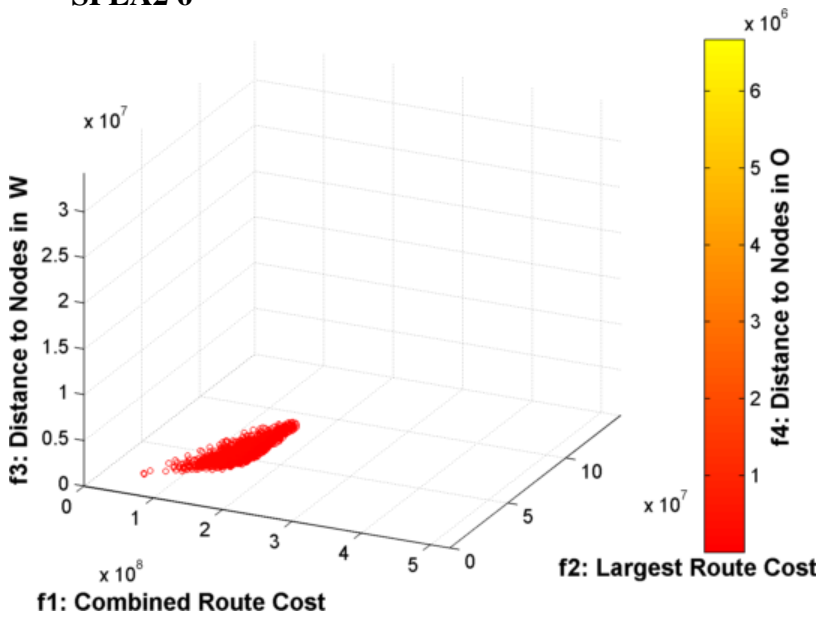
SPEA2 4



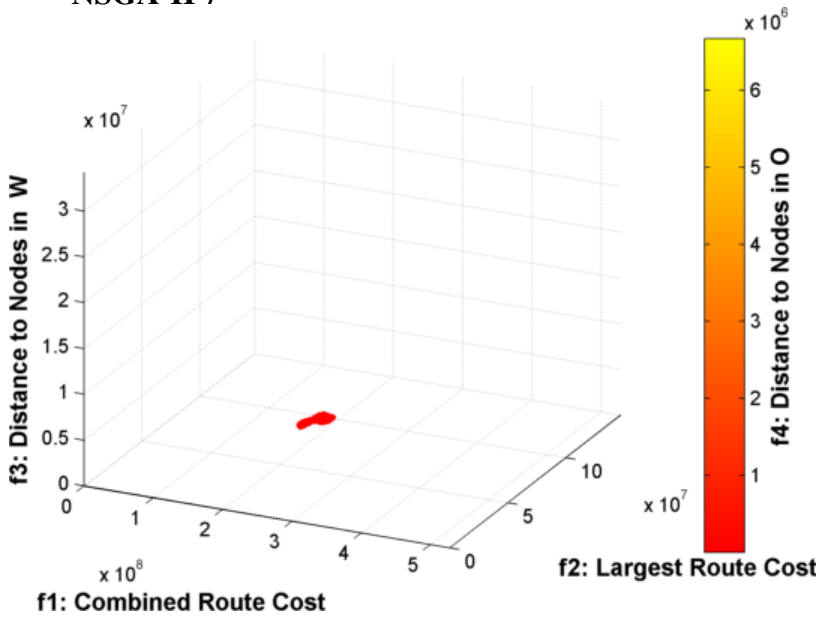
SPEA2 5



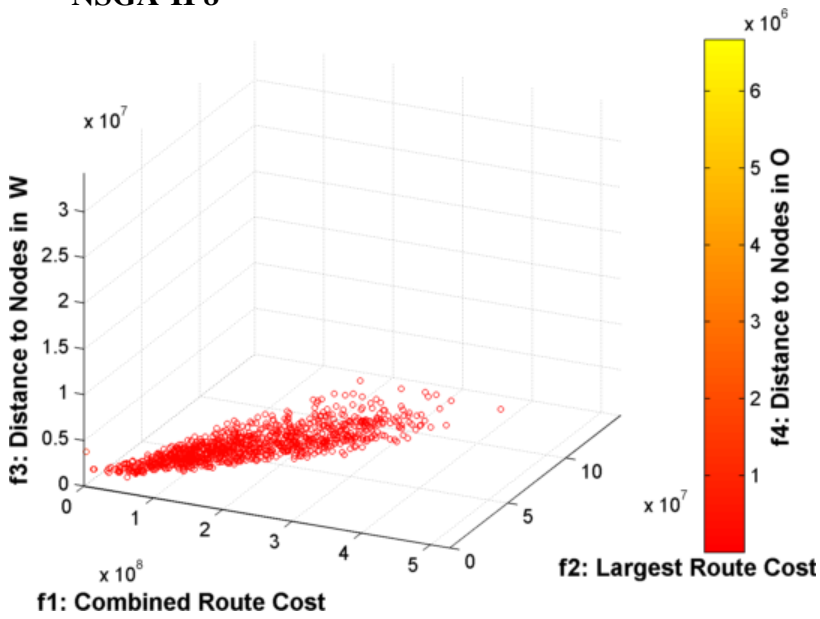
SPEA2 6



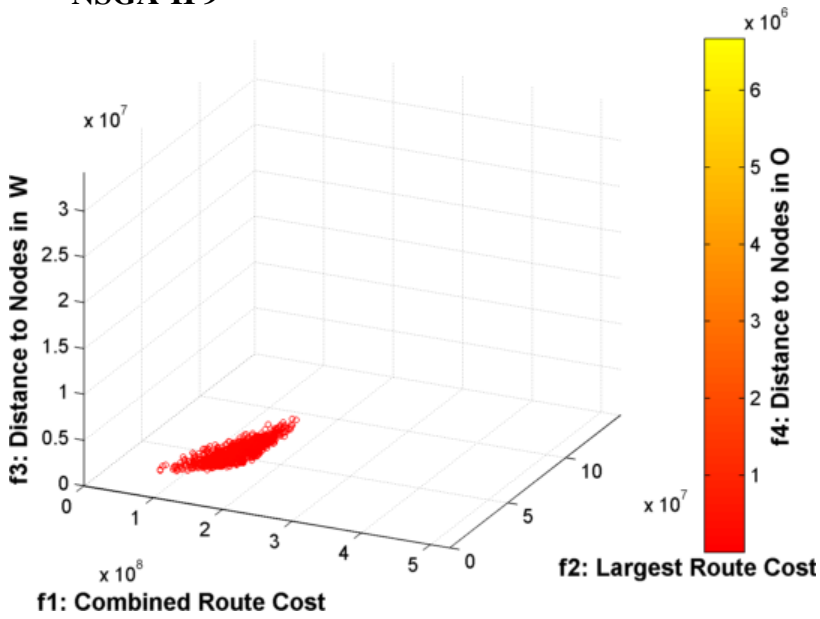
NSGA-II 7



NSGA-II 8



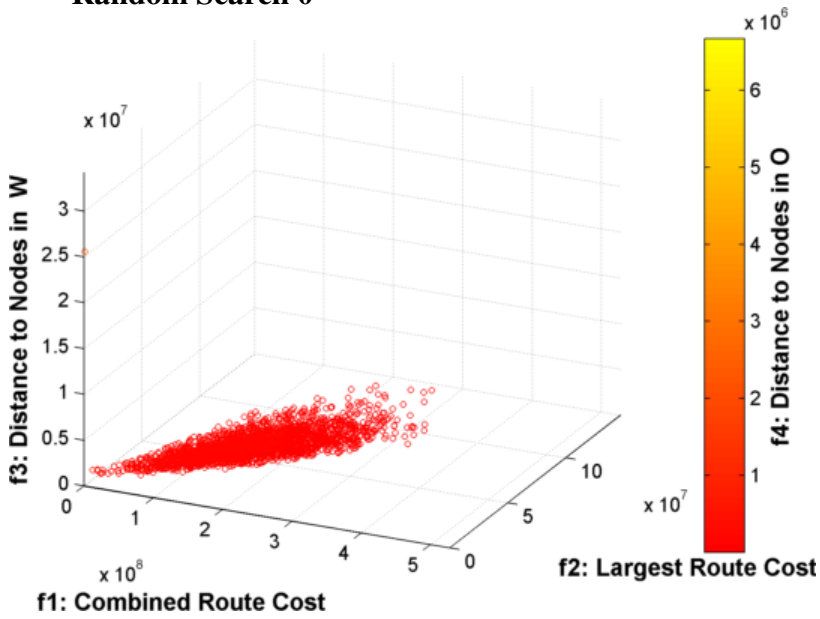
NSGA-II 9



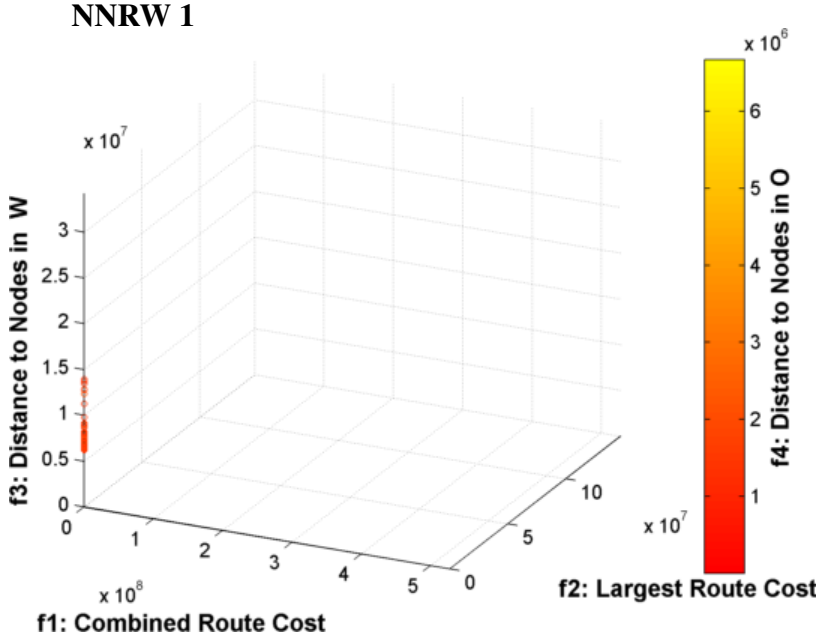
B.3 Canada

B.3.1 Moderate EDOD.

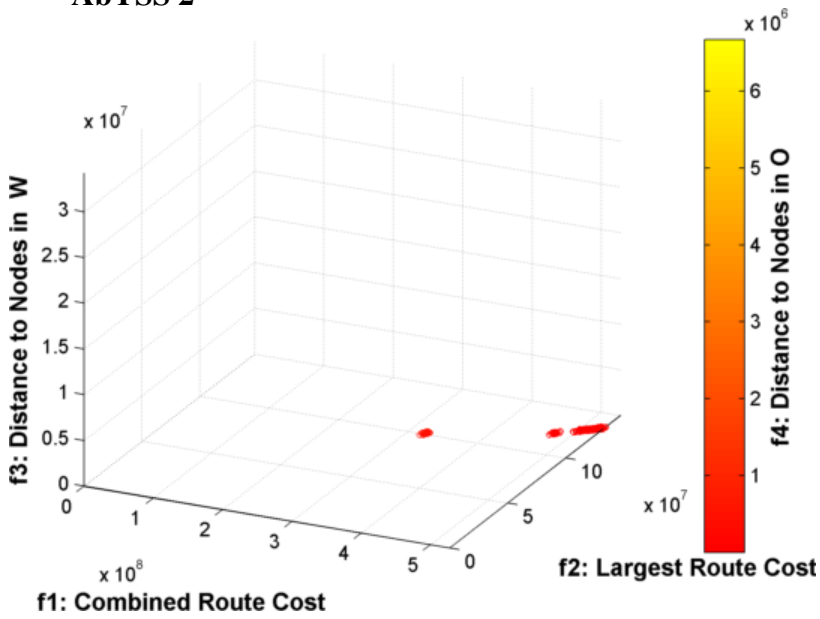
Random Search 0



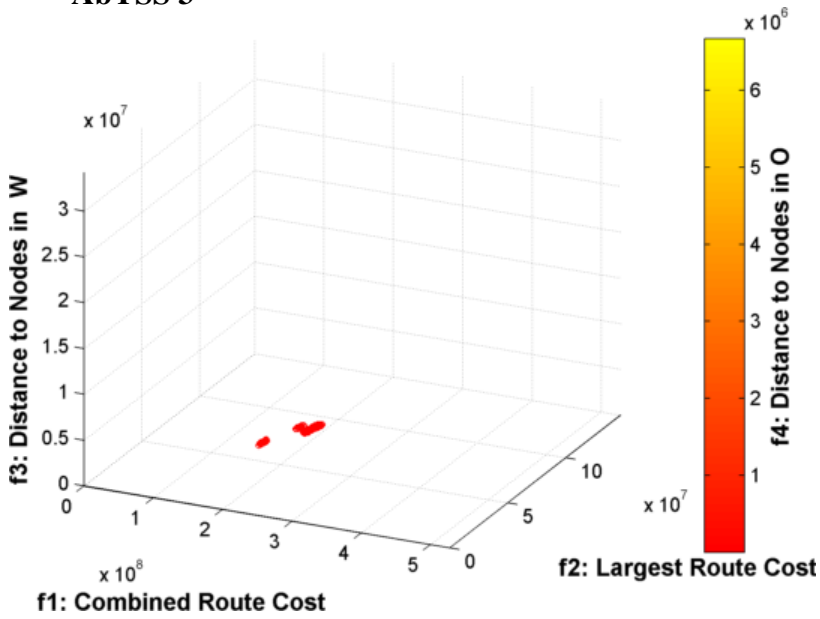
NNRW 1



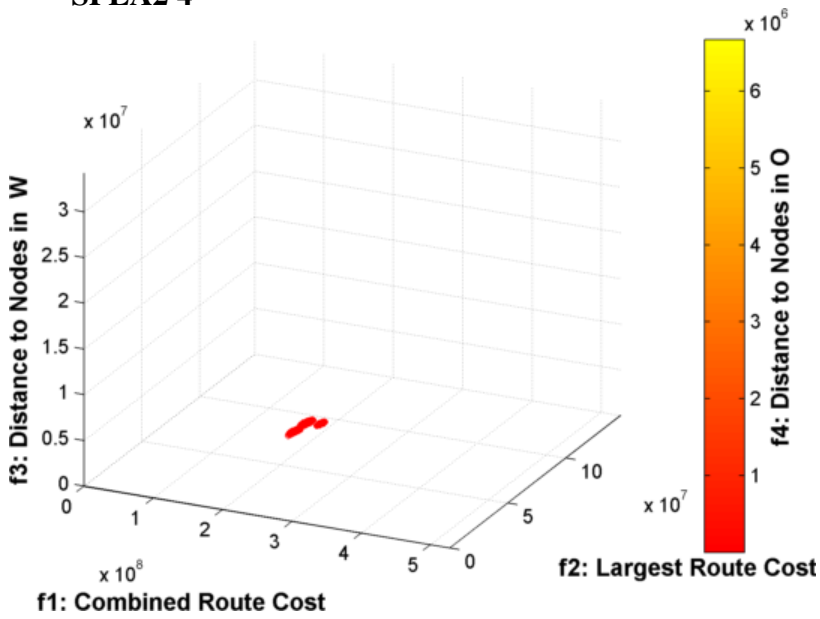
AbYSS 2



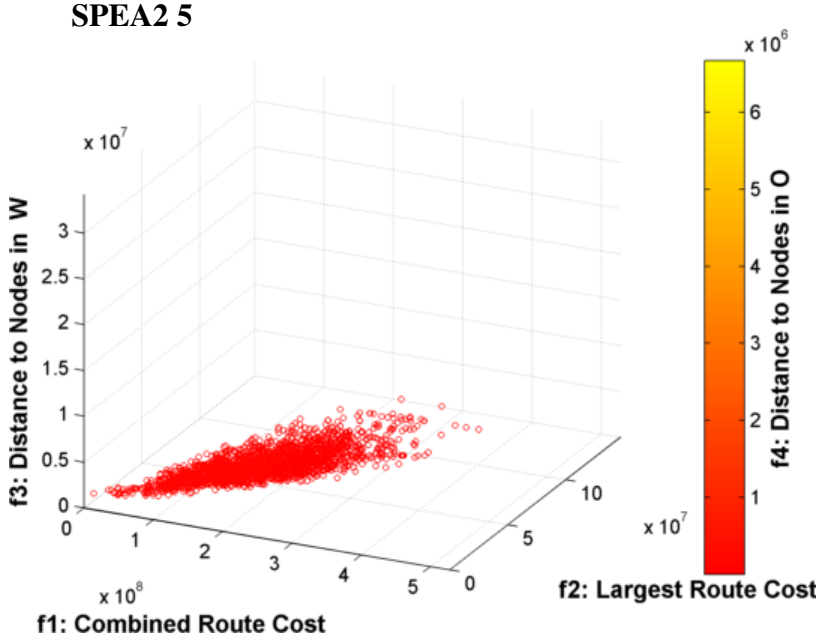
AbYSS 3



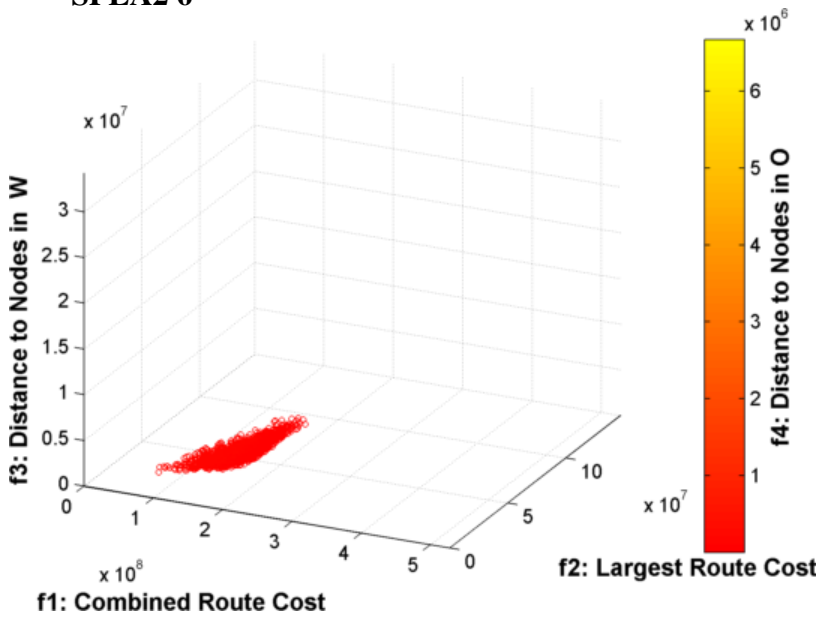
SPEA2 4



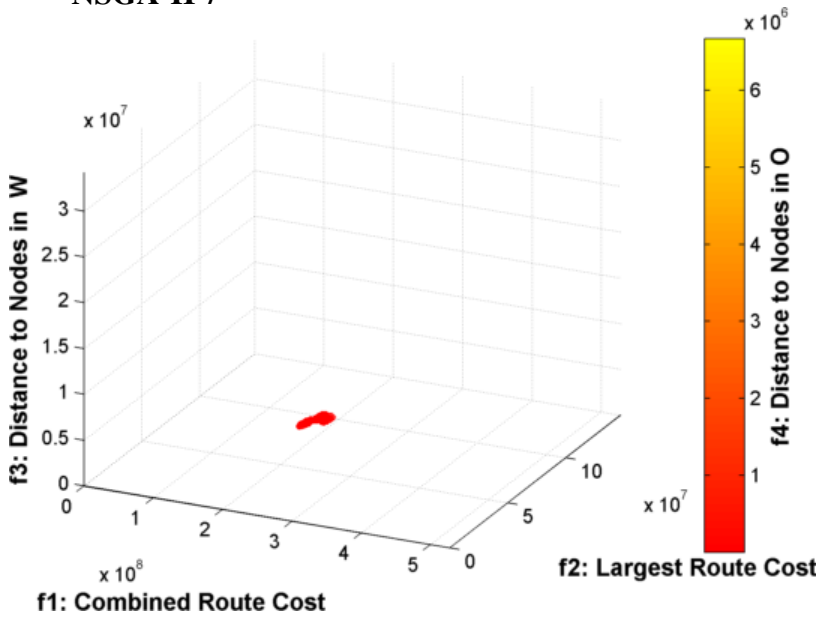
SPEA2 5



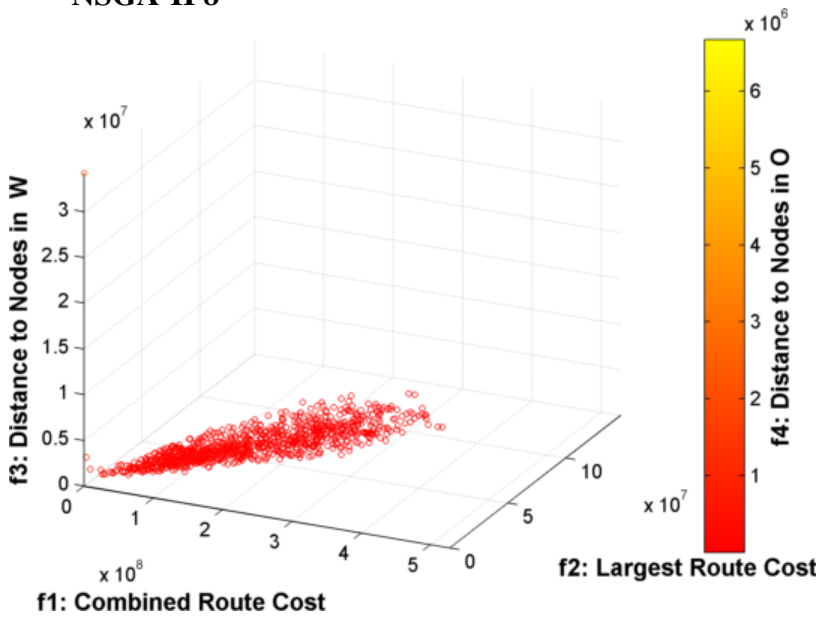
SPEA2 6



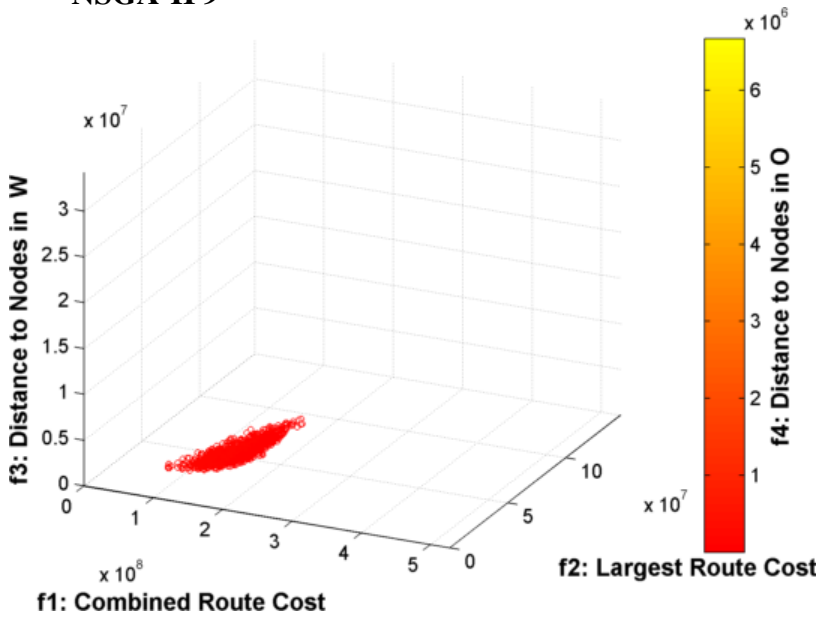
NSGA-II 7



NSGA-II 8



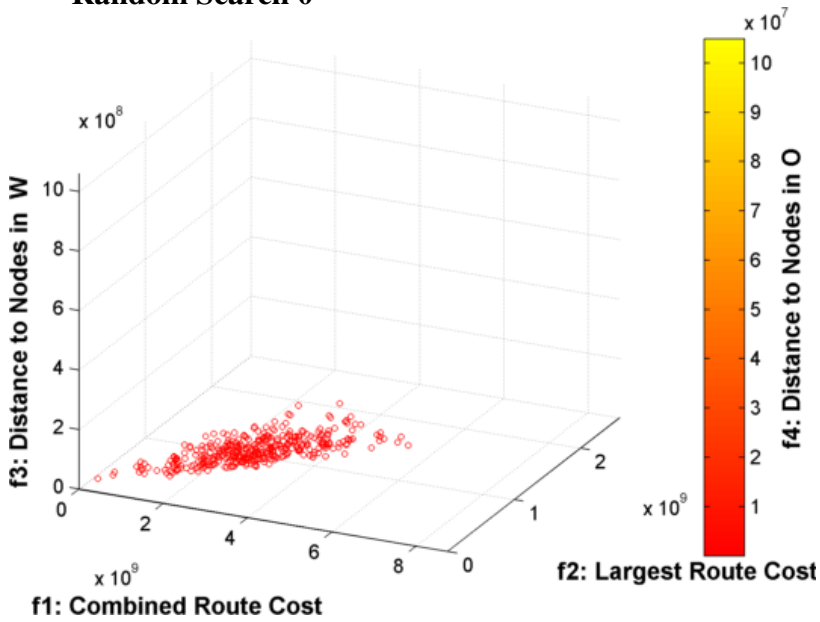
NSGA-II 9



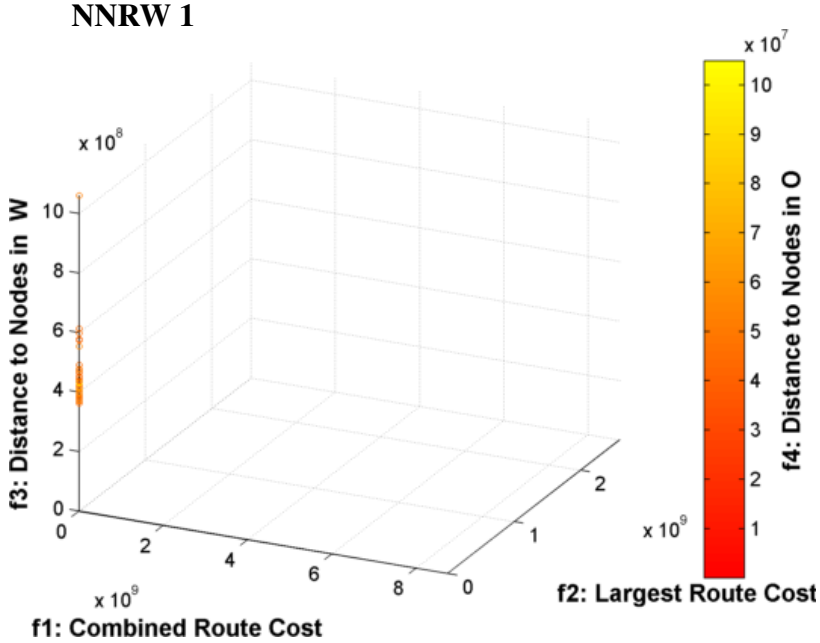
B.4 USA

B.4.1 Light EDOD.

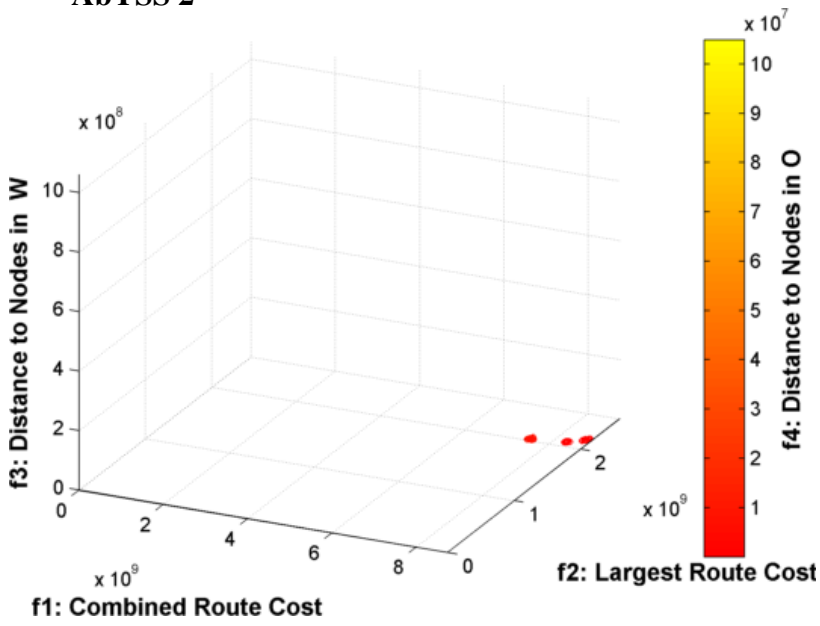
Random Search 0



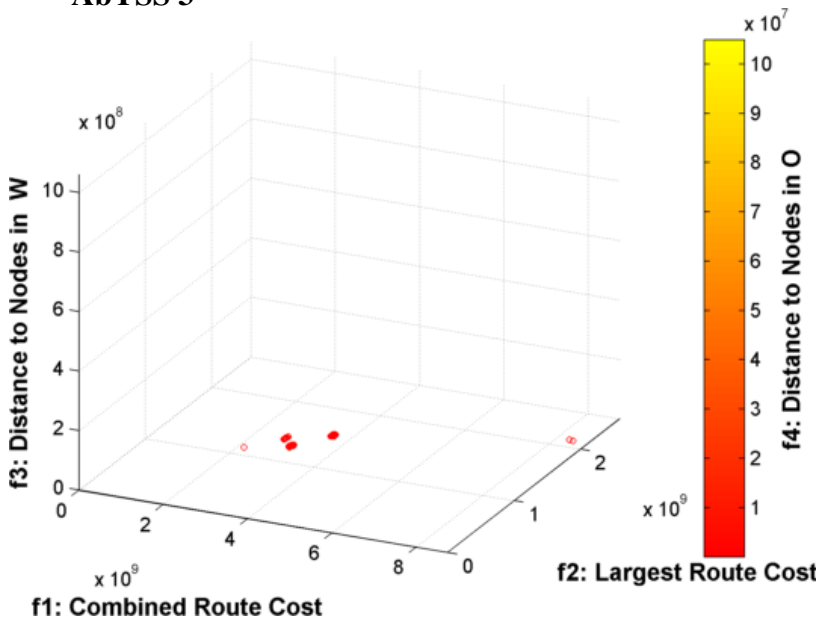
NNRW 1



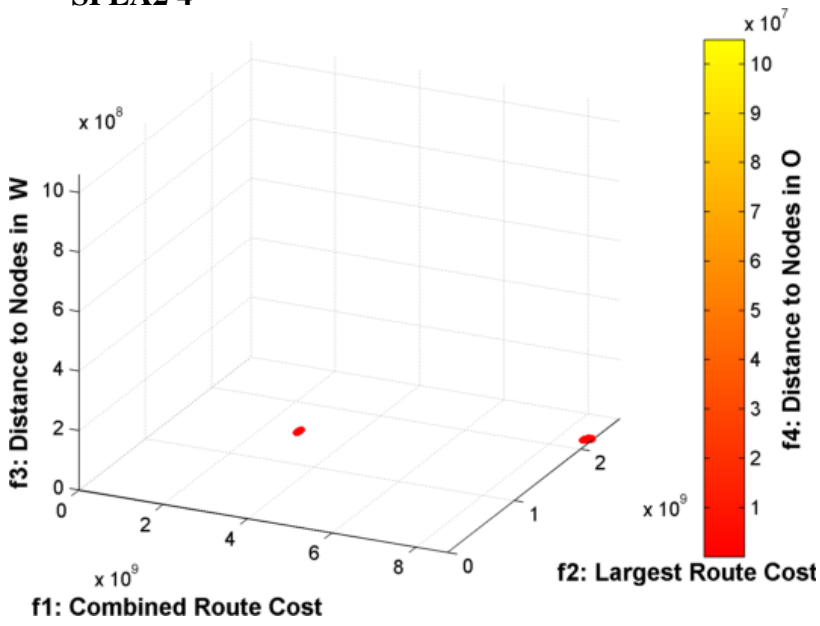
AbYSS 2



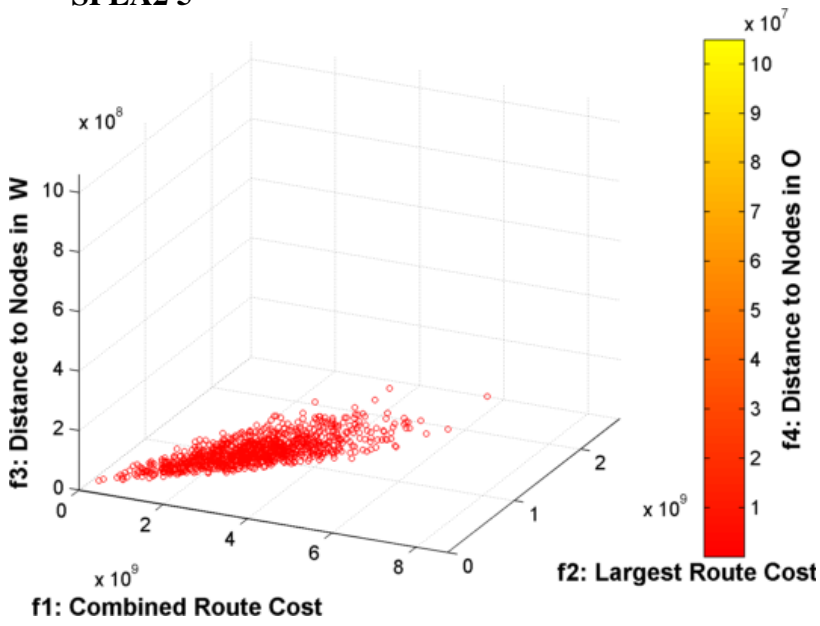
AbYSS 3



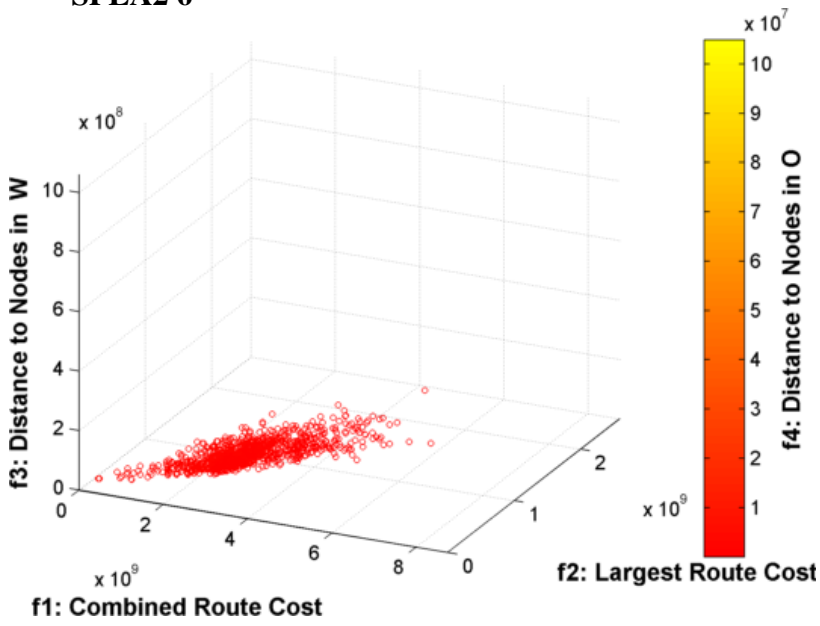
SPEA2 4



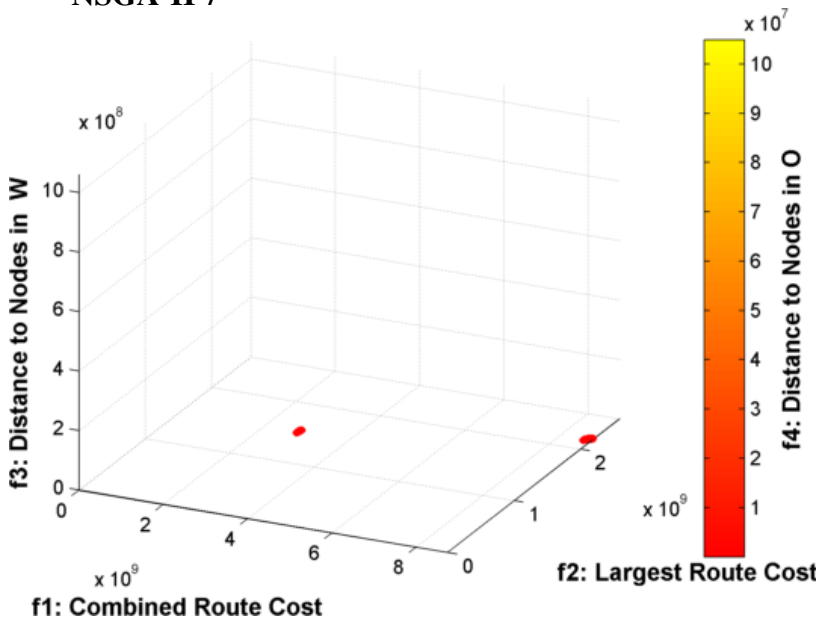
SPEA2 5



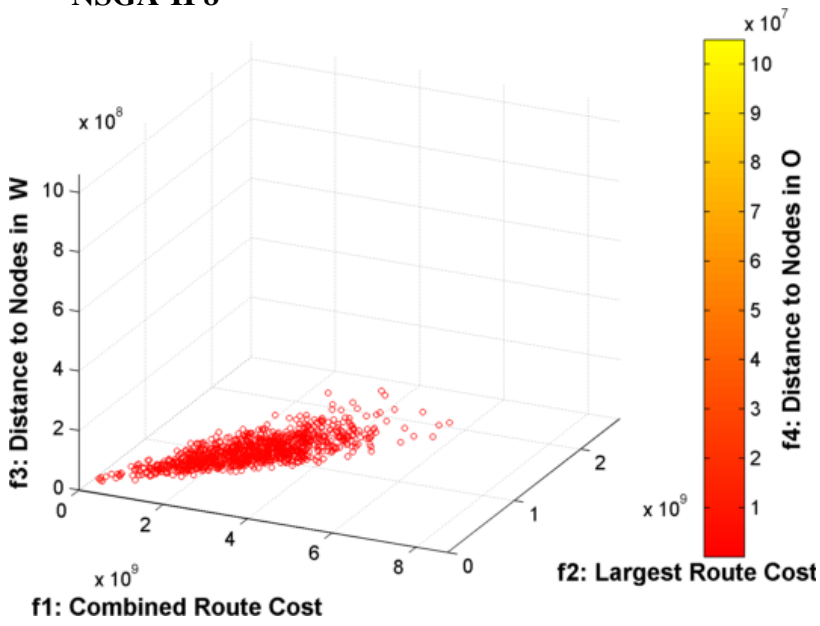
SPEA2 6



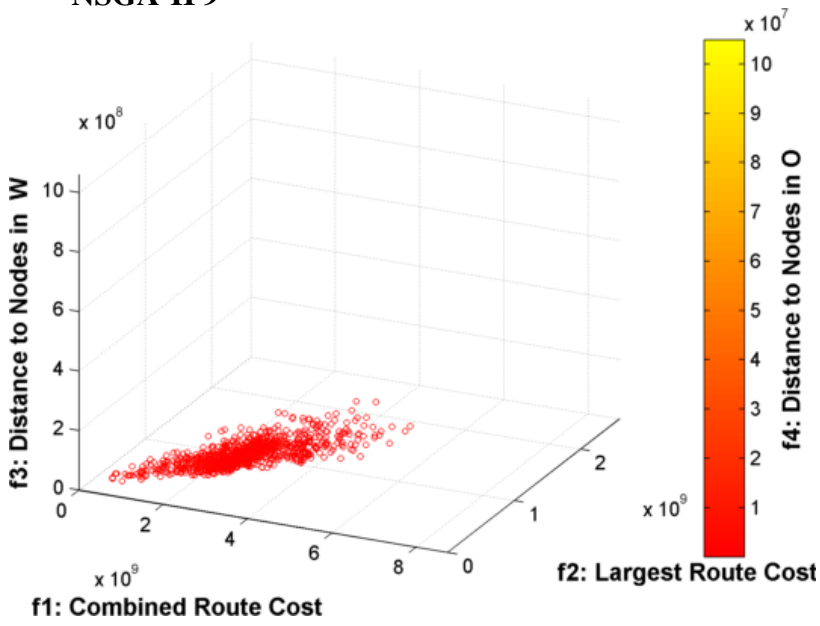
NSGA-II 7



NSGA-II 8

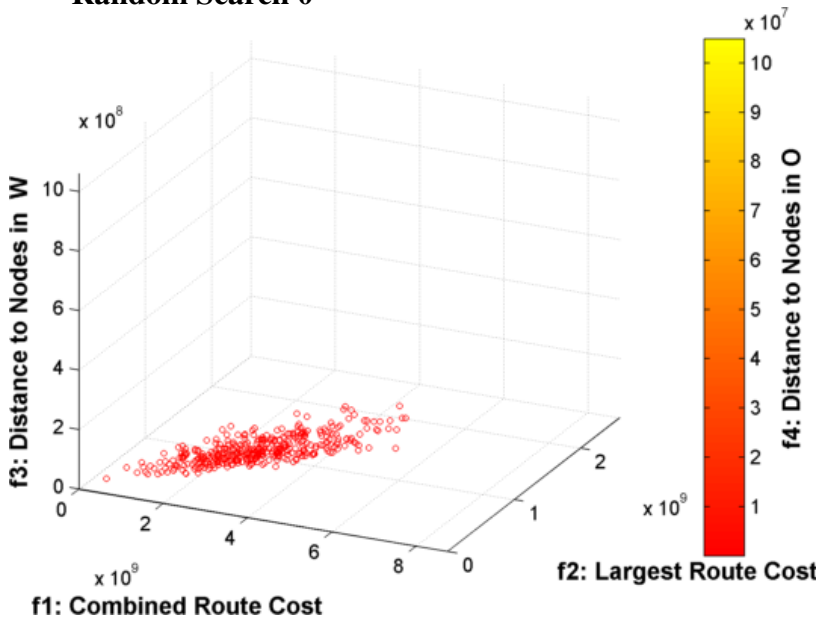


NSGA-II 9

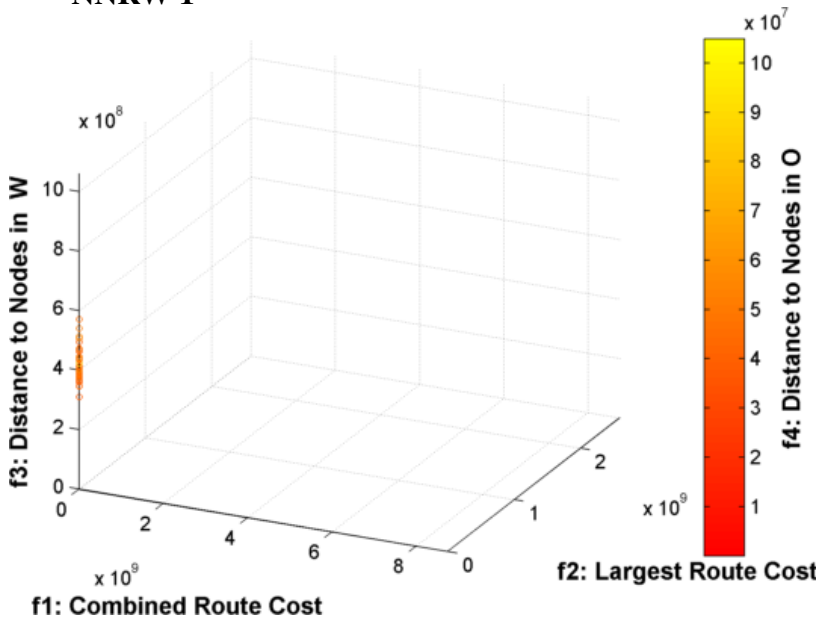


B.4.2 Moderate EDOD.

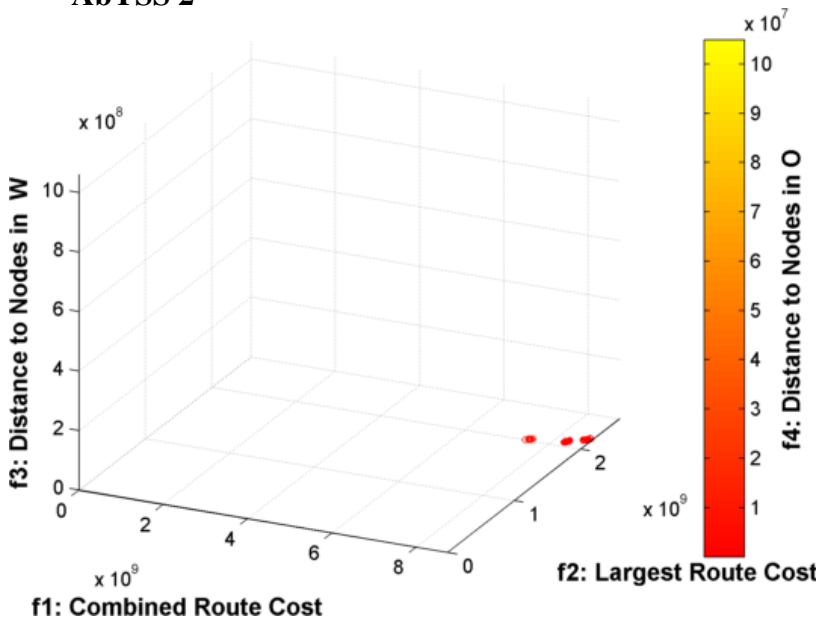
Random Search 0



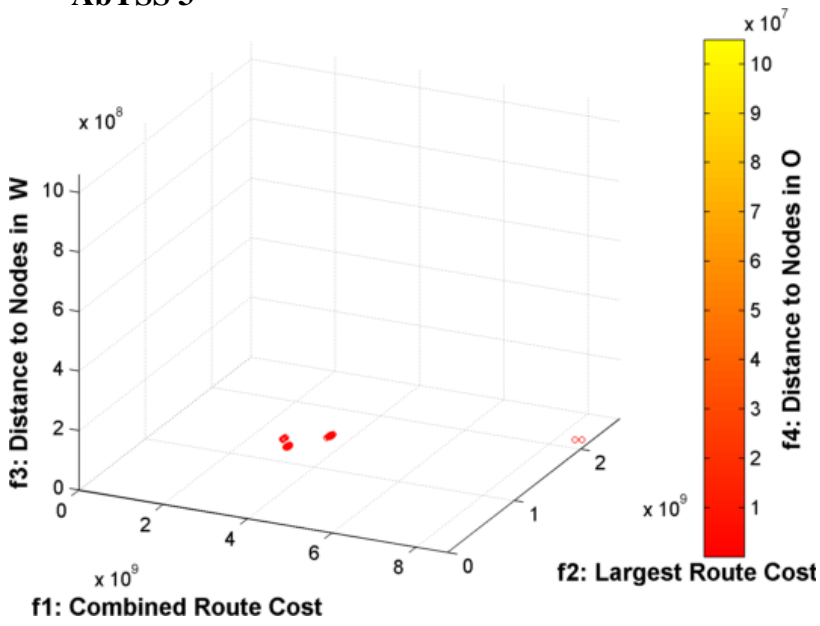
NNRW 1



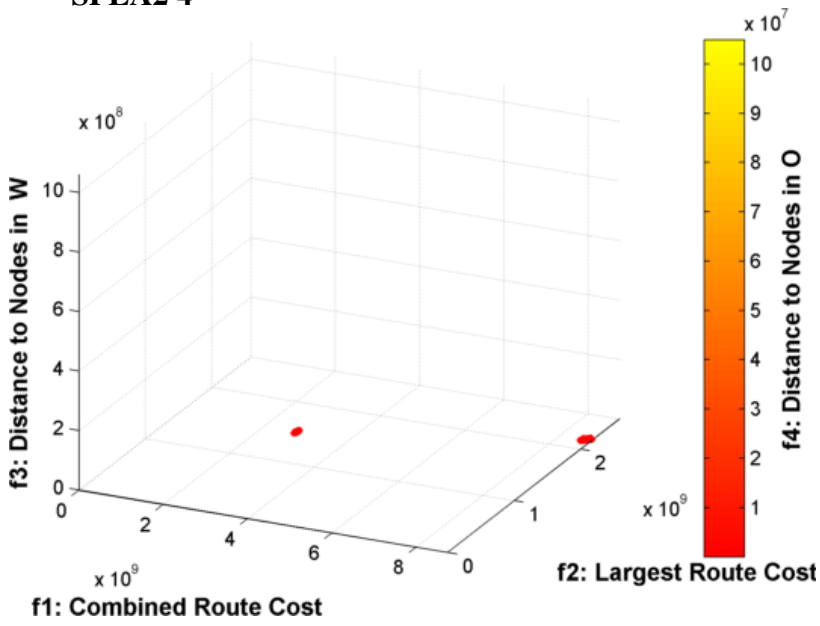
AbYSS 2



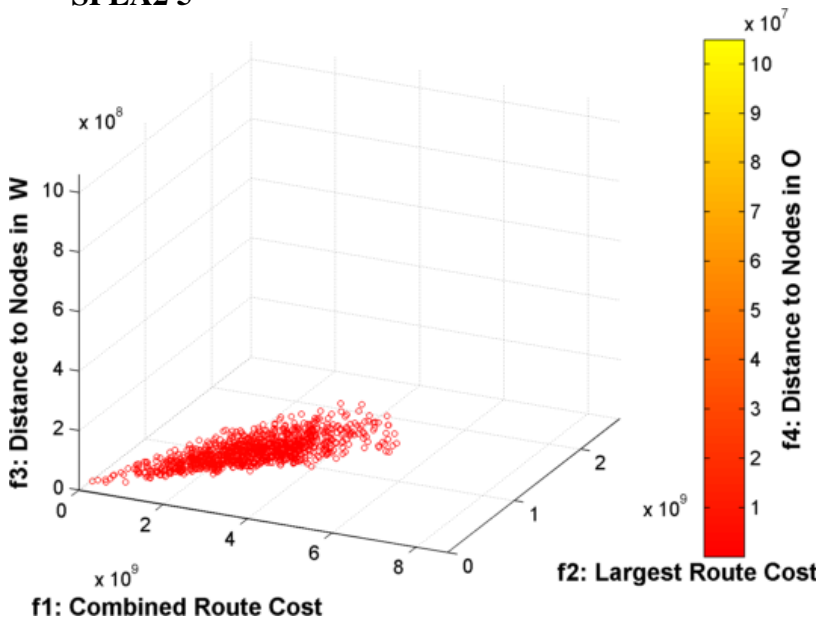
AbYSS 3



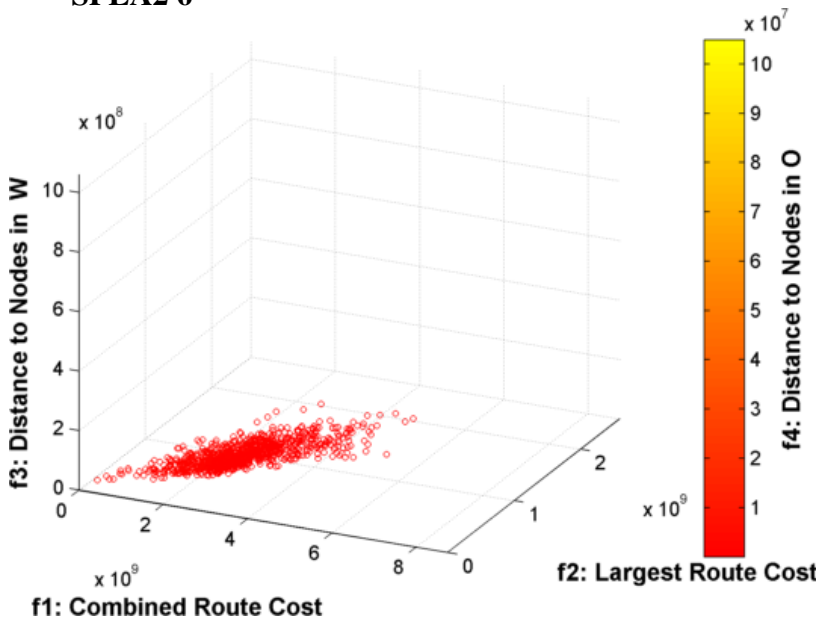
SPEA2 4



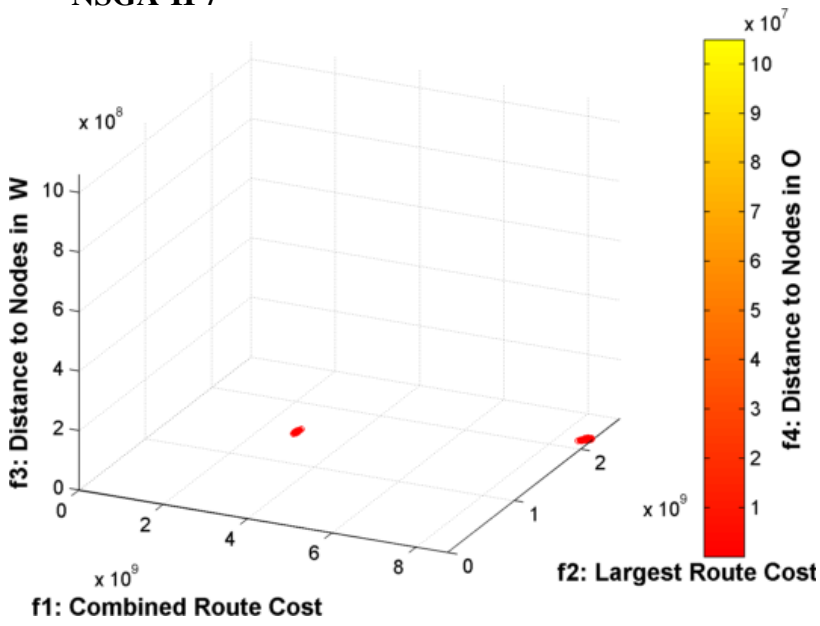
SPEA2 5



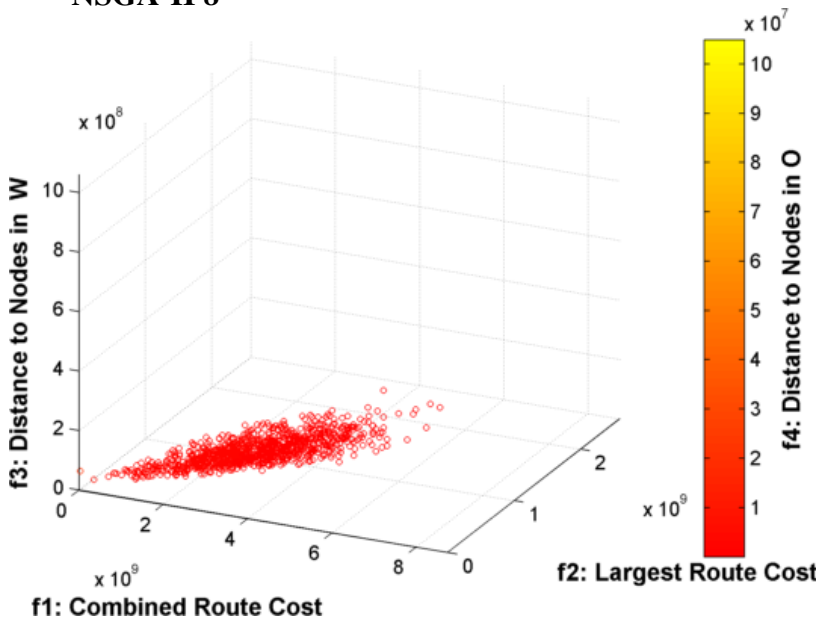
SPEA2 6



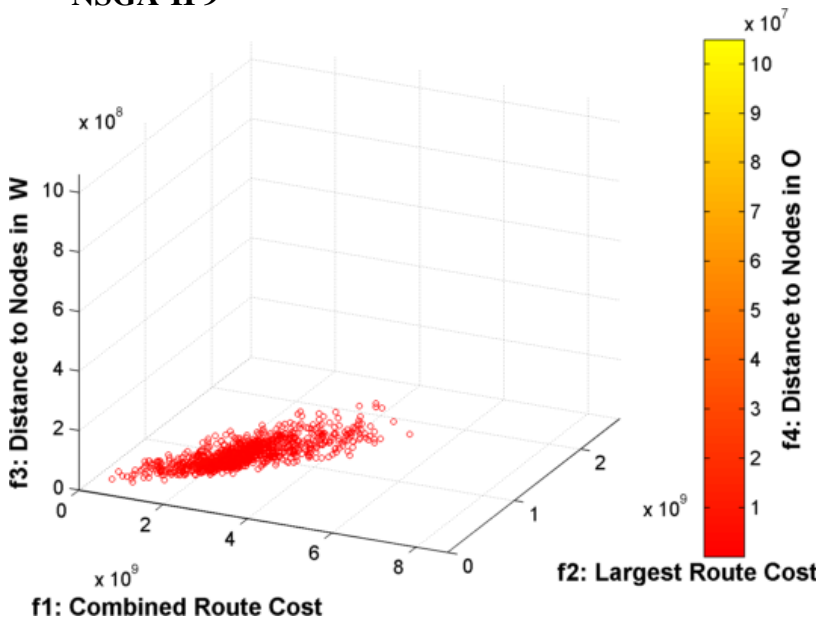
NSGA-II 7



NSGA-II 8



NSGA-II 9



Bibliography

[1] "Amazon EC2 Instance Types". http://aws.amazon.com/ec2/instance-types/. Accessed: 2013, April 1.

[2] "Factsheets: MQ-1 Predator". http://www.af.mil/information/factsheets/factsheet.asp?fsID=122. Accessed: 2013, April 1.

[3] "Factsheets: MQ-9 Reaper". http://www.af.mil/information/factsheets/factsheet.asp?fsID=6405. Accessed: 2013, April 1.

[4] "Lynx / AN/APY-8". http://www.defense-update.com/products/l/lynx-sar.htm. Accessed: 2013, April 1.

[5] "Lynx Synthetic Aperture Radar". http://www.sandia.gov/radar/lynx.html. Accessed: 2013, April 1.

[6] "National Traveling Salesman Problem". http://www.tsp.gatech.edu/world/countries.html. Accessed: 2012, December 10.

[7] "TSPLIB". http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/. Accessed: 2012, December 10.

[8] Abbaspour, Rahim A. and Farhad Samadzadegan. "Time-Dependent Personal Tour Planning and Scheduling in Metropolises". *Expert Systems with Applications*, 38(10):12439–12452, 2011.

[9] Ahr, Dino and Gerhard Reinelt. "A Tabu Search Algorithm for the Min–Max k–Chinese Postman Problem". *Computers & Operations Research*, 33(12):3403–3422, 2006.

[10] Arsie, Alessandro, Ketan Savla, and Emilio Frazzoli. "Efficient Routing Algorithms for Multiple Vehicles with no Explicit Communications". *Automatic Control, IEEE Transactions on*, 54(10):2302–2317, 2009.

[11] Attanasio, Andrea, Jean-François Cordeau, Gianpaolo Ghiani, and Gilbert Laporte. "Parallel Tabu Search Heuristics for the Dynamic Multi-Vehicle Dial-a-Ride Problem". *Parallel Computing*, 30(3):377–387, 2004.

[12] Awerbuch, Baruch, Yossi Azar, Avrim Blum, and Santosh Vempala. "New Approximation Guarantees for Minimum-Weight k-Trees and Prize-Collecting Salesmen". *SIAM Journal on Computing*, 28(1):254–262, 1998.

[13] Awerbuch, Baruch, Margrit Betke, Ronald L Rivest, and Mona Singh. "Piecemeal Graph Exploration by a Mobile Robot". *Information and Computation*, 152(2):155–172, 1999.

[14] Bäck, Thomas, D. B. Fogel, and T. Michalewicz. "Introduction to Evolutionary Algorithms". *Evolutionary Computation*, 1, 2000.

[15] Balas, Egon. "The Prize Collecting Traveling Salesman Problem". *Networks*, 19(6):621–636, 1989.

[16] Baldacci, Roberto, Paolo Toth, and Daniele Vigo. "Recent Advances in Vehicle Routing Exact Algorithms". *4OR: A Quarterly Journal of Operations Research*, 5(4):269–298, December 2007. ISSN 1619-4500. URL http://dx.doi.org/10.1007/s10288-007-0063-3.

[17] Barceló, Jaime, Hanna Grzybowska, and Sara Pardo. "Vehicle Routing and Scheduling Models, Simulation and City Logistics". *Dynamic Fleet Management*, 163–195, 2007.

[18] Beaudry, Alexandre, Gilbert Laporte, Teresa Melo, and Stefan Nickel. "Dynamic Transportation of Patients in Hospitals". *OR spectrum*, 32(1):77–107, 2010.

[19] Bent, Russell and Pascal Van Hentenryck. "Regrets Only! Online Stochastic Optimization Under Time Constraints". *Proceedings of the National Conference on Artificial Intelligence*, 501–506. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

[20] Bent, Russell and Pascal Van Hentenryck. "Online Stochastic and Robust Optimization". *Advances in Computer Science-ASIAN 2004. Higher-Level Decision Making*, 3237–3238, 2005.

[21] Bent, Russell W and Pascal Van Hentenryck. "Scenario-Based Planning for Partially Dynamic Vehicle Routing with Stochastic Customers". *Operations Research*, 52(6):977–987, 2004.

[22] Bertsimas, Dimitris J. and David Simchi-Levi. "A New Generation of Vehicle Routing Research: Robust Algorithms, Addressing Uncertainty". *Operations Research*, 44(2):286–304, 1996.

[23] Bertsimas, Dimitris J. and Garrett Van Ryzin. "A Stochastic and Dynamic Vehicle Routing Problem in the Euclidean Plane". *Operations Research*, 39(4):601–615, 1991.

[24] Bertsimas, Dimitris J. and Garrett Van Ryzin. "Stochastic and Dynamic Vehicle Routing in the Euclidean Plane with Multiple Capacitated Vehicles". *Operations Research*, 41(1):60–76, 1993.

[25] Bertsimas, Dimitris J. and Garrett Van Ryzin. "Stochastic and Dynamic Vehicle Routing with General Demand and Interarrival Time Distributions". *Advances in Applied Probability*, 947–978, 1993.

[26] Blasco, X., J.M. Herrero, J. Sanchis, and M. Martínez. "A New Graphical Visualization of n–Dimensional Pareto Front for Decision-Making in Multiobjective Optimization". *Information Sciences*, 178(20):3908–3924, 2008.

[27] Bowerman, Robert, Brent Hall, and Paul Calamai. "A Multi-Objective Optimization Approach to Urban School Bus Routing: Formulation and Solution Method". *Transportation Research Part A: Policy and Practice*, 29(2):107–123, 1995.

[28] Bräysy, Olli and Michel Gendreau. "Vehicle Routing Problem with Time Windows, Part I: Route construction and Local Search Algorithms". *Transportation science*, 39(1):104–118, 2005.

[29] Brown, Darin T. *Routing Unmanned Aerial Vehicles While Considering General Restricted Operating Zones*. Master's thesis, 2001.

[30] Bullo, Francesco, Emilio Frazzoli, Marco Pavone, Ketan Savla, and Stephen L Smith. "Dynamic Vehicle Routing for Robotic Systems". *Proceedings of the IEEE*, 99(9):1482–1504, 2011.

[31] Chao, I., Bruce L. Golden, Edward A Wasil, et al. "The Team Orienteering Problem". *European journal of operational research*, 88(3):464–474, 1996.

[32] Chen, Huey-Kuo, Che-Fu Hsueh, and Mei-Shiang Chang. "The Real-Time Time-Dependent Vehicle Routing Problem". *Transportation Research Part E: Logistics and Transportation Review*, 42(5):383–408, 2006.

[33] Chevrier, Rémy, Arnaud Liefooghe, Laetitia Jourdan, and Clarisse Dhaenens. "Solving a Dial-a-Ride Problem with a Hybrid Evolutionary Multi-Objective Approach: Application to Demand Responsive Transport". *Applied Soft Computing*, 2011.

[34] Chini, Giorgia, Guido Oddi, and Antonio Pietrabissa. "An Adaptive Cooperative Receding Horizon Controller for the Multivehicle routing Problem". *Department of Computer, Control, and Management Engineering Antonio Ruberti Technical Reports*, 1(8), 2013.

[35] Chiu, Ko-Ming and Jing-Sin Liu. "Robot Routing Using Clustering-Based Parallel Genetic Algorithm with Migration". *Merging Fields Of Computational Intelligence And Sensor Technology (CompSens), 2011 IEEE Workshop On*, 42–49. IEEE, 2011.

[36] Choi, Eunjeong and Dong-Wan Tcha. "A Column Generation Approach to the Heterogeneous Fleet Vehicle Routing Problem". *Computers & Operations Research*, 34(7):2080–2095, 2007.

[37] Christofides, Nicos, Aristide Mingozzi, and Paolo Toth. "Exact Algorithms for the Vehicle Routing Problem, Based on Spanning Tree and Shortest Path Relaxations". *Mathematical programming*, 20(1):255–282, 1981.

[38] Coello, Carlos A. Coello, Gary B Lamont, and David A Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, 2007.

[39] Cordeau, Jean-François, Gilbert Laporte, Martin W. P. Savelsbergh, and Daniele Vigo. *Chapter 6 Vehicle Routing*, volume 14, 367–428. Elsevier, 2007. ISBN 9780444513465. URL http://dx.doi.org/10.1016/s0927-0507(06)14006-2.

[40] Cordeau, Jean-Francois, Guy Desaulniers, Jacques Desrosiers, Marius M Solomon, and François Soumis. "VRP with Time Windows". *The vehicle routing problem*, 9:157–193.

[41] Cordeau, Jean-François, Michel Gendreau, Gilbert Laporte, Jean-Yves Potvin, and François Semet. "A Guide to Vehicle Routing Heuristics". *Journal of the Operational Research Society*, 512–522, 2002.

[42] Crainic, Teodor G., Michel Gendreau, and Jean-Yves Potvin. "Intelligent Freight-Transportation Systems: Assessment and the Contribution of Operations Research". *Transportation Research Part C: Emerging Technologies*, 17(6):541–557, December 2009. ISSN 0968090X. URL http://dx.doi.org/10.1016/j.trc.2008.07.002.

[43] Crino, John R. *A Group Theoretic Tabu search Methodology for Solving the Theater Distribution Vehicle Routing and Scheduling Problem*. Ph.D. thesis, 2002.

[44] Current, John R and David A Schilling. "The Covering Salesman Problem". *Transportation Science*, 23(3):208–213, 1989.

[45] Dantzig, G. B. and J. H. Ramser. "The Truck Dispatching Problem". *Management Science*, 6(1):80–91, 1959. URL http://dx.doi.org/10.1287/mnsc.6.1.80.

[46] Deb, Kalyanmoy, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II". *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.

[47] Desaulniers, Guy, June Lavigne, and Francois Soumis. "Multi-Depot Vehicle Scheduling Problems with Time Windows and Waiting Costs". *European Journal of Operational Research*, 111(3):479–494, 1998.

[48] Dias, M. Bernardine, Robert Zlot, Nidhi Kalra, and Anthony Stentz. "Market-Based Multirobot Coordination: A Survey and Analysis". *Proceedings of the IEEE*, 94(7):1257–1270, 2006.

[49] Doerner, Karl, Axel Focke, and Walter J. Gutjahr. "Multicriteria Tour Planning for Mobile Healthcare Facilities in a Developing Country". *European Journal of Operational Research*, 179(3):1078–1096, 2007. ISSN 03772217. URL http://dx.doi.org/10.1016/j.ejor.2005.10.067.

[50] Doerner, Karl F., Walter J. Gutjahr, Richard F. Hartl, Michaela Karall, and Marc Reimann. "Heuristic Solution of an Extended Double-Coverage Ambulance Location Problem for Austria". *Central European Journal of Operations Research*, 13(4):325–340, 2005.

[51] Doumit, Sarjoun and Ali Minai. "Distributed Resource Exploitation for Autonomous Mobile Sensor Agents in Dynamic Environments". *Unifying Themes in Complex Systems*, 1:511, 2008.

[52] Durillo, Juan J. and Antonio J. Nebro. "jMetal: A Java Framework for Multi-Objective Optimization". *Advances in Engineering Software*, 42:760–771, 2011. ISSN 0965-9978. URL http://www.sciencedirect.com/science/article/pii/S0965997811001219.

[53] Enright, John J., Ketan Savla, Emilio Frazzoli, and Francesco Bullo. "Stochastic and Dynamic Routing Problems for Multiple Uninhabited Aerial Vehicles". 2009.

[54] Eyckelhof, Casper and Marko Snoek. "Ant Systems for a Dynamic TSP". *Ant Algorithms*, 88–99, 2002.

[55] Feillet, Dominique, Pierre Dejax, and Michel Gendreau. "Traveling Salesman Problems with Profits". *Transportation science*, 39(2):188–205, 2005.

[56] Feo, Thomas A. and Mauricio G.C. Resende. "Greedy Randomized Adaptive Search Procedures". *Journal of global optimization*, 6(2):109–133, 1995.

[57] Fleischmann, Bernhard, Stefan Gnutzmann, and Elke Sandvoß. "Dynamic Vehicle Routing Based on Online Traffic Information". *Transportation science*, 38(4):420–433, 2004.

[58] Fonseca, Carlos M., Peter J. Fleming, et al. "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization". *Proceedings of the Fifth International Conference on Genetic Algorithms*, volume 1, 416. San Mateo, California, 1993.

[59] Frazzoli, E. and F. Bullo. "Decentralized Algorithms for Vehicle Routing in a Stochastic Time-Varying Environment". *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 4, 3357–3363. 2004. URL http://dx.doi.org/10.1109/CDC.2004.1429220.

[60] Frazzoli, John J. Enright Emilio and Marco Pavone Ketan Savla. *UAV Routing and Coordination in Stochastic, Dynamic Environments*.

[61] Fu, Si-Yao, Li-Wei Han, Yu Tian, and Guo-Sheng Yang. "Path Planning for Unmanned Aerial Vehicle Based on Genetic Algorithm". *Cognitive Informatics & Cognitive Computing (ICCI* CC), 2012 IEEE 11th International Conference on*, 140–144. IEEE, 2012.

[62] Garcia-Najera, Abel and John A. Bullinaria. "An Improved Multi-Objective Evolutionary Algorithm for the Vehicle Routing Problem with Time Windows". *Computers & Operations Research*, 38(1):287–300, 2011.

[63] Gendreau, Michel, Francois Guertin, Jean-Yves Potvin, and Eric Taillard. "Parallel Tabu Search for Real-Time Vehicle Routing and Dispatching". *Transportation science*, 33(4):381–390, 1999.

[64] Gendreau, Michel, Gilbert Laporte, and René Séguin. "Stochastic Vehicle Routing". *European Journal of Operational Research*, 88(1):3–12, 1996.

[65] Gendreau, Michel, Gilbert Laporte, and Frédéric Semet. "The Covering Tour Problem". *Operations Research*, 45(4):568–576, July 1997. ISSN 1526-5463. URL http://dx.doi.org/10.1287/opre.45.4.568.

[66] Gendreau, Michel and Jean-Yves Potvin. *Dynamic Vehicle Routing and Dispatching*. Springer, 1998.

[67] Ghiani, Gianpaolo, Francesca Guerriero, Gilbert Laporte, and Roberto Musmanno. "Real-Time Vehicle Routing: Solution Concepts, Algorithms and Parallel Computing Strategies". *European Journal of Operational Research*, 151(1):1–11, 2003.

[68] Ghiani, Gianpaolo, Emanuele Manni, and Barrett W. Thomas. "A Comparison of Anticipatory Algorithms for the Dynamic and Stochastic Traveling Salesman Problem". *Transportation Science*, 46(3):374–387, 2012.

[69] Ghiani, Gianpaolo, Antonella Quaranta, and Chefi Triki. "New Policies for the Dynamic Traveling Salesman Problem". *Optimisation Methods and Software*, 22(6):971–983, 2007.

[70] Goel, Asvin and Volker Gruhn. "A General Vehicle Routing Problem". *European Journal of Operational Research*, 191(3):650–660, 2008.

[71] Güner, Ali R., Alper Murat, and Ratna Babu Chinnam. "Dynamic Routing Under Recurrent and Non-Recurrent Congestion Using Real-time ITS Information". *Computers & Operations Research*, 39(2):358–373, 2012.

[72] Ha, Min Hoang, Nathalie Bostel, André Langevin, and Louis-Martin Rousseau. "Solving the Close-Enough Arc Routing Problem". 2012.

[73] Hachicha, Mondher, M John Hodgson, Gilbert Laporte, and Frederic Semet. "Heuristics for the Multi-Vehicle Covering Tour Problem". *Computers and Operations research*, 27(1):29–42, 2000.

[74] Haghani, Ali and Soojung Jung. "A Dynamic Vehicle Routing Problem with Time-Dependent Travel Times". *Computers & operations research*, 32(11):2959–2986, 2005.

[75] van Hemert, Jano and J La Poutré. "Dynamic Routing Problems with Fruitful Regions: Models and Evolutionary Computation". *Parallel Problem Solving from Nature-PPSN VIII*, 692–701. Springer, 2004.

[76] Herbawi, Wesam and Michael Weber. "Ant Colony vs. Genetic Multiobjective Route Planning in Dynamic Multi-hop Ridesharing". *Tools with Artificial Intelligence (ICTAI), 2011 23rd IEEE International Conference on*, 282–288. IEEE, 2011.

[77] Herbawi, Wesam and Michael Weber. "Comparison of Multiobjective Evolutionary Algorithms for Solving the Multiobjective Route Planning in Dynamic Multi-Hop Ridesharing". *Evolutionary Computation (CEC), 2011 IEEE Congress on*, 2099–2106. IEEE, 2011.

[78] Hodgson, M. John, Gilbert Laporte, and Frederic Semet. "A Covering Tour Model for Planning Mobile Health Care Facilities in Suhum District, Ghama". *Journal of Regional Science*, 38(4):621–638, November 1998. ISSN 0022-4146. URL http://dx.doi.org/10.1111/0022-4146.00113.

[79] Hvattum, Lars M., Arne Løkketangen, and Gilbert Laporte. "Solving a Dynamic and Stochastic Vehicle Routing Problem with a Sample Scenario Hedging Heuristic". *Transportation Science*, 40(4):421–438, 2006.

[80] Hvattum, Lars Magnus, Arne Løkketangen, and Gilbert Laporte. "A Branch-and-Regret Heuristic for Stochastic and Dynamic Vehicle Routing Problems". *Networks*, 49(4):330–340, 2007.

[81] Ichoua, Soumia, Michel Gendreau, and Jean-Yves Potvin. "Exploiting Knowledge About Future Demands for Real-Time Vehicle Dispatching". *Transportation Science*, 40(2):211–225, 2006.

[82] Ichoua, Soumia, Michel Gendreau, and Jean-Yves Potvin. "Planned Route Optimization for Real-Time Vehicle Routing". *Dynamic Fleet Management*, 1–18, 2007.

[83] Irani, Sandy, Xiangwen Lu, and Amelia Regan. "On-line Algorithms for the Dynamic Traveling Repair Problem". *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 517–524. Society for Industrial and Applied Mathematics, 2002.

[84] James, F. "Monte Carlo Theory and Practice". *Reports on Progress in Physics*, 43(9):1145, 1980.

[85] Jozefowiez, Nicolas, Frédéric Semet, and El-Ghazali Talbi. "The Bi-Objective Covering Tour Problem". *Computers & operations research*, 34(7):1929–1942, 2007.

[86] Jozefowiez, Nicolas, Frederic Semet, and El-Ghazali Talbi. "Multi-Objective Vehicle Routing Problems". *European Journal of Operational Research*, 189(2):293–309, 2008.

[87] Karp, Richard M. "Complexity of Computer Computations", 1972.

[88] Lagoudakis, M., Evangelos Markakis, David Kempe, Pinar Keskinocak, Anton Kleywegt, Sven Koenig, Craig Tovey, Adam Meyerson, and Sonal Jain. "Auction-Based Multi-Robot Routing". *Robotics: Science and Systems*, 343–350. MIT Press, 2005.

[89] Laporte, Gilbert. "What You Should Know About the Vehicle Routing Problem". *Naval Research Logistics (NRL)*, 54(8):811–819, 2007.

[90] Laporte, Gilbert. "Fifty Years of Vehicle Routing". *Transportation Science*, 43(4):408–416, 2009.

[91] Larsen, Allan, Oli B.G. Madsen, and M. Solomon. "Partially Dynamic Vehicle Routing-Models and Algorithms". *Journal of the Operational Research Society*, 637–646, 2002.

[92] Larsen, Allan, Oli B.G. Madsen, and Marius M. Solomon. "The a priori Dynamic Traveling Salesman Problem with Time Windows". *Transportation Science*, 38(4):459–472, 2004.

[93] Larsen, Allan, Oli B.G. Madsen, and Marius M. Solomon. "Recent Developments in Dynamic Vehicle Routing Systems". *The Vehicle Routing Problem: Latest Advances and New Challenges*, 199–218, 2008.

[94] Lawler, Eugene L, Jan Karel Lenstra, A.H.G. Rinnooy Kan, and David B. Shmoys. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, volume 3. Wiley New York, 1985.

[95] Li, Changhe, Ming Yang, and Lishan Kang. "A New Approach to Solving Dynamic Traveling Salesman Problems". *Simulated Evolution and Learning*, 236–243, 2006.

[96] Li, Jing-Quan, Pitu B. Mirchandani, and Denis Borenstein. "A Lagrangian Heuristic for the Real-Time Vehicle Rescheduling Problem". *Transportation Research Part E: Logistics and Transportation Review*, 45(3):419–433, 2009.

[97] Li, Jing-Quan, Pitu B. Mirchandani, and Denis Borenstein. "Real-Time Vehicle Rerouting Problems with Time Windows". *European Journal of Operational Research*, 194(3):711–727, 2009.

[98] Li, LYO and Z. Fu. "The School Bus Routing Problem: A Case Study". *Journal of the Operational Research Society*, 552–558, 2002.

[99] Li, Wei and Christos G. Cassandras. "A Cooperative Receding Horizon Controller for Multivehicle Uncertain Environments". *Automatic Control, IEEE Transactions on*, 51(2):242–257, 2006.

[100] Li, Weiqi and Mingyuan Feng. "A Parallel Procedure for Dynamic Multi-objective TSP". *2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications (ISPA)*, 1–8. 2012.

[101] Lorini, Sandro, Jean-Yves Potvin, and Nicolas Zufferey. "Online Vehicle Routing and Scheduling with Dynamic Travel Times". *Computers & Operations Research*, 38(7):1086–1090, 2011.

[102] Lund, Karsten, Oli B.G. Madsen, and Jens M. Rygaard. *Vehicle Routing Problems with Varying Degrees of Dynamism*. IMM Institute of Mathematical Modelling, 1996.

[103] Madani, Omid, Steve Hanks, and Anne Condon. "On the Undecidability of Probabilistic Planning and Infinite-Horizon Partially Observable Markov Decision Problems". *Proceedings of the National Conference on Artificial Intelligence*, 541–548. John Wiley & Sons LTD, 1999.

[104] Madsen, Oli B.G., Hans F. Ravn, and Jens Moberg Rygaard. "A Heuristic Algorithm for a Dial-a-Ride Problem with Time Windows, Multiple Capacities, and Multiple Objectives". *Annals of operations Research*, 60(1):193–208, 1995.

[105] Magnanti, Thomas L. "Combinatorial Optimization and Vehicle Fleet Planning: Perspectives and Prospects". *Networks*, 11(2):179–213, 1981.

[106] Manni, Emanuele. "Topics in Real-Time Fleet Management". *4OR: A Quarterly Journal of Operations Research*, 7(2):203–206, 2009.

[107] Mavrovouniotis, Michalis and Shengxiang Yang. "A Memetic Ant Colony Optimization Algorithm for the Dynamic Travelling Salesman Problem". *Soft Computing-A Fusion of Foundations, Methodologies and Applications*, 15(7):1405–1425, 2011.

[108] Mei, Hao, Yantao Tian, and Linan Zu. "A Hybrid Ant Colony Optimization Algorithm for Path Planning of Robot in Dynamic Environment". *International Journal of Information Technology*, 12(3):78–88, 2006.

[109] Meignan, David, Jean-Charles Creput, and Abderrafiâa Koukam. "A Coalition-Based Metaheuristic for the Vehicle Routing Problem". *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, 1176–1182. IEEE, 2008.

[110] Mes, Martijn, Matthieu Van Der Heijden, and Aart Van Harten. "Comparison of Agent-Based Scheduling to Look-Ahead Heuristics for Real-Time Transportation Problems". *European Journal of Operational Research*, 181(1):59–75, 2007.

[111] Michalewicz, Zbigniew and David B. Fogel. *How to Solve It: Modern Heuristics*. Springer, 2004.

[112] Mingozzi, Aristide, Simone Giorgi, and Roberto Baldacci. "An Exact Method for the Vehicle Routing Problem with Backhauls". *Transportation Science*, 33(3):315–329, 1999.

[113] Mitrović-Minić, Snežana and Gilbert Laporte. "Waiting Strategies for the Dynamic Pickup and Delivery Problem with Time Windows". *Transportation Research Part B: Methodological*, 38(7):635–655, 2004.

[114] Mosteo, Alejandro R., Luis Montano, and Michail G. Lagoudakis. "Multi-Robot Routing Under Limited Communication Range". *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, 1531–1536. IEEE, 2008.

[115] Mu, Q., Z. Fu, Jens Lysgaard, and R. Eglese. "Disruption Management of the Vehicle Routing Problem with Vehicle Breakdown". *Journal of the Operational Research Society*, 62(4):742–749, 2010.

[116] Mu, Qianxin and Richard W. Eglese. "Disrupted Capacitated Vehicle Routing Problem with Order Release Delay". *Annals of Operations Research*, 1–16, 2011.

[117] Mukai, Naoto, Toyohide Watanabe, and Jun Feng. "Proactive Route Planning Based on Expected Rewards for Transport Systems". *Tools with Artificial Intelligence, 2005. ICTAI 05. 17th IEEE International Conference on*, 7–pp. IEEE, 2005.

[118] Murata, Tadahiko and Ryota Itai. "Multi-Objective Vehicle Routing Problems Using Two-Fold EMO Algorithms to Enhance Solution Similarity on Non-Dominated Solutions". *Evolutionary Multi-Criterion Optimization*, 885–896. Springer, 2005.

[119] Nebro, Antonio J., Francisco Luna, Enrique Alba, Bernabé Dorronsoro, Juan J. Durillo, and Andreas Beham. "AbYSS: Adapting Scatter Search to Multiobjective Optimization". *Evolutionary Computation, IEEE Transactions on*, 12(4):439–457, 2008.

[120] Novoa, Clara and Robert Storer. "An Approximate Dynamic Programming Approach for the Vehicle Routing Problem with Stochastic Demands". *European Journal of Operational Research*, 196(2):509–515, 2009.

[121] Novoa, Clara M. *Static and Dynamic Approaches for Solving the Vehicle Routing Problem with Stochastic Demands*. Lehigh University, 2006.

[122] O'Rourke, Kevin P., William B. Carlton, T. Glenn Bailey, and Raymond R. Hill. "Dynamic Routing of Unmanned Aerial Vehicles Using Reactive Tabu Search". *Military Operations Research*, 6(1):5–30, 2001.

[123] Papadimitriou, Christos H. and John N. Tsitsiklis. "The Complexity of Markov Decision Processes". *Mathematics of operations research*, 12(3):441–450, 1987.

[124] Park, Junhyuk and Byung-In Kim. "The School Bus Routing Problem: A Review". *European Journal of operational research*, 202(2):311–319, 2010.

[125] Pavone, Marco. *Dynamic Vehicle Routing for Robotic Networks*. Ph.D. thesis, Massachusetts Institute of Technology, 2010.

[126] Pillac, Victor, Michel Gendreau, Christelle Guéret, and Andrés L. Medaglia. "A Review of Dynamic Vehicle Routing Problems". *European Journal of Operational Research*, 225(1):1–11, 2011. ISSN 03772217. URL http://dx.doi.org/10.1016/j.ejor.2012.08.015.

[127] Pineau, Joelle, Geoff Gordon, Sebastian Thrun, et al. "Point-Based Value Iteration: An Anytime Algorithm for POMDPs". *International joint conference on artificial intelligence*, volume 18, 1025–1032. Lawrence Erlabaum Associates LTD, 2003.

[128] Popa, Dan O., Arthur C. Sanderson, Rick J. Komerska, Sai S. Mupparapu, D. Richard Blidberg, and Steven G. Chappel. "Adaptive Sampling Algorithms for Multiple Autonomous Underwater Vehicles". *Autonomous Underwater Vehicles, 2004 IEEE/OES*, 108–118. IEEE, 2004.

[129] Potvin, Jean-Yves, Ying Xu, and Ilham Benyahia. "Vehicle Routing and Scheduling with Dynamic Travel Times". *Computers & Operations Research*, 33(4):1129–1137, 2006.

[130] Psaraftis, Harilaos N. "A Dynamic Programming Solution to the Single Vehicle Many-to-Many Immediate Request Dial-a-Ride Problem". *Transportation Science*, 14(2):130–154, 1980.

[131] Ravassi, Santiago. *Analysis of the Dynamic Traveling Salesman Problem with Different Policies*. Ph.D. thesis, Concordia University, 2011.

[132] Reinelt, Gerhard. "TSPLIB - A Traveling Salesman Problem Library". *ORSA journal on computing*, 3(4):376–384, 1991.

[133] Ropke, Stefan and David Pisinger. "A Unified Heuristic for a Large Class of Vehicle Routing Problems with Backhauls". *European Journal of Operational Research*, 171(3):750–775, 2006.

[134] Roy, Nicholas, Geoffrey J. Gordon, and Sebastian Thrun. "Finding Approximate POMDP Solutions Through Belief Compression". *Journal of Artificial Intelligence Research*, 23:1–40, 2005.

[135] Russell, Stuart Jonathan, Peter Norvig, Ernest Davis, Stuart Jonathan Russell, and Stuart Jonathan Russell. *Artificial Intelligence: A Modern Approach*. Prentice hall Upper Saddle River, NJ, 2010.

[136] Schaffer, J. David. "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms". *Proceedings of the 1st international Conference on Genetic Algorithms*, 93–100. L. Erlbaum Associates Inc., 1985.

[137] Secomandi, Nicola. "Comparing Neuro-Dynamic Programming Algorithms for the Vehicle Routing Problem with Stochastic Demands". *Computers & Operations Research*, 27(11):1201–1225, 2000.

[138] Secomandi, Nicola and François Margot. "Reoptimization Approaches for the Vehicle-Routing Problem with Stochastic Demands". *Operations research*, 57(1):214–230, 2009.

[139] Sessomboon, W., K. Watanabe, T. Irohara, and K. Yoshimoto. "A Study on Multi-Objective Vehicle Routing Problem Considering Customer Satisfaction with Due-Time (the Creation of Pareto Optimal Solutions by Hybrid Genetic Algorithm)". *Trans. Jpn. Soc. Mech. Eng*, 64:1108–1115, 1998.

[140] Shin, Hyung Sik and Sanjay Lall. "An Approximate Dynamic Programming Approach to the Dynamic Traveling Repairperson Problem". *Decision and Control (CDC), 2010 49th IEEE Conference on*, 2286–2291. IEEE, 2010.

[141] Smith, Stephen L., Marco Pavone, Francesco Bullo, and Emilio Frazzoli. "Dynamic Traveling Repairperson with Priority Demands". *Center for Control, Dynamical Systems and Computation. University of California at Santa Barbara, Tech. Rep*, 600:08–0904, 2008.

[142] Soares, Marcelo B., Mario F.M. Campos, Dimas A. Dutra, V.C. da S. Campos, and Guilherme A.S. Pereira. "Hybrid Mobile Robot Navigational Strategy for Efficient Data Collection in Sparsely Deployed Sensor Networks". *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 2833–2838. IEEE, 2007.

[143] Sörensen, Kenneth and Marc Sevaux. "A Practical Approach for Robust and Flexible Vehicle Routing Using Metaheuristics and Monte Carlo Sampling". *Journal of Mathematical Modelling and Algorithms*, 8(4):387–407, 2009.

[144] Sungur, Ilgaz, Fernando Ordónez, and Maged M. Dessouky. "A Robust Optimization Approach for the Capacitated Vehicle Routing Problem with Demand Uncertainty". *University of Southern California, Working paper*, 6:2006, 2006.

[145] Tagmouti, Mariam, Michel Gendreau, and Jean-Yves Potvin. "Arc Routing Problems with Time-Dependent Service Costs". *European Journal of Operational Research*, 181(1):30–39, 2007.

[146] Tan, Huang Teng. *The In-Transit Vigilant Covering Tour Problem for Routing Unmanned Ground Vehicles*. Master's thesis, AFIT, 2012.

[147] Tan, Kay Chen, Y.H. Chew, and Loo Hay Lee. "A Hybrid Multi-Objective Evolutionary Algorithm for Solving Truck and Trailer Vehicle Routing Problems". *European Journal of Operational Research*, 172(3):855–885, 2006.

[148] Taniguchi, Eiichi and Hiroshi Shimamoto. "Intelligent Transportation System Based Dynamic Vehicle Routing and Scheduling with Variable Travel Times". *Transportation Research Part C: Emerging Technologies*, 12(3):235–250, 2004.

[149] Tao, Guo and Zbigniew Michalewicz. "Inver-Over Operator for the TSP". *Parallel Problem Solving from Nature–PPSN V*, 803–812. Springer, 1998.

[150] of Technology. Urban Systems Laboratory, Massachusetts Institute and Nigel H.M. Wilson. *Scheduling Algorithms for a Dial-a-Ride System*. 1971.

[151] Thomas, Barrett W. "Waiting Strategies for Anticipating Service Requests from Known Customer Locations". *Transportation science*, 41(3):319–331, 2007.

[152] van de Voorde, A. *The Online UAV Mission Planning Problem*. Master's thesis, University of Utrecht, Sep 2012.

[153] Waisanen, Holly A, Devavrat Shah, and Munther A Dahleh. "A Dynamic Pickup and Delivery Problem in Mobile Networks Under Information Constraints". *Automatic Control, IEEE Transactions on*, 53(6):1419–1433, 2008.

[154] Wilson, Nigel H.M. and Neil J. Colvin. *Computer Control of the Rochester Dial-a-Ride System*. Massachusetts Institute of Technology, Center for Transportation Studies, 1977.

[155] Wilson, Nigel H.M., Richard Wayne Weissberg, and John Hauser. *Advanced Dial-a-Ride Algorithms Research Project*. Technical report, 1976.

[156] Wolfler Calvo, R. and R. Cordone. "A Heuristic Approach to the Overnight Security Service Problem". *Computers & Operations Research*, 30(9):1269–1287, 2003.

[157] Yang, Ming, Zhou Kang, and Lishan Kang. "A Parallel Multi-algorithm Solver for Dynamic Multi-Objective TSP (DMO-TSP)". *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, 164–173, 2008.

[158] Yang, Xin-She. *Engineering Optimization: An Introduction with Metaheuristic Applications*. Wiley, 2010.

[159] Yuan, Bo, Maria Orlowska, and Shazia Sadiq. "On the Optimal Robot Routing Problem in Wireless Sensor Networks". *Knowledge and Data Engineering, IEEE Transactions on*, 19(9):1252–1261, 2007.

[160] Zitzler, Eckart, Marco Laumanns, and Lothar Thiele. "SPEA2: Improving the Strength Pareto Evolutionary Algorithm", 2001.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
13–06–2013	Master's Thesis	Oct 2010–June 2013

4. TITLE AND SUBTITLE

The Dynamic Multi-objective Multi-vehicle Covering Tour Problem

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Ziegler, Joshua S.,

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB, OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-13-J-09

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Intentionally Left Blank

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES

This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

This work introduces a new routing problem called the Dynamic Mulit-Objective Multi-vehicle Covering Tour Problem (DMOMCTP). The DMOMCTPs is a combinatorial optimization problem that represents the problem of routing multiple vehicles to survey an area in which unpredictable target nodes may appear during execution. The formulation includes multiple objectives that include minimizing the cost of the combined tour cost, minimizing the longest tour cost, minimizing the distance to nodes to be covered and maximizing the distance to hazardous nodes. This study adapts several existing algorithms to the problem with several operator and solution encoding variations. The efficacy of this set of solvers is measured against six problem instances created from existing Traveling Salesman Problem instances which represent several real countries. The results indicate that repair operators, variable length solution encodings and variable-length operators obtain a better approximation of the true Pareto front.

15. SUBJECT TERMS

Add four or five key words/phrases for indexing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Gilbert L. Peterson (ENG)
U	U	U	UU	170	19b. TELEPHONE NUMBER *(include area code)* (937) 785-6565 x4281 gilbert.peterson@afit.edu